Adaptive Weight-Shift Flight Control of Flexible Wing Aircraft

by

Nathaniel Mailhot

Thesis submitted to the University of Ottawa
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
in
Mechanical Engineering

Examining Committee

The following served on the examining committee for this thesis. They rendered a verdict of minor revisions and nominated the thesis for a University of Ottawa PhD thesis prize.

External Member: Jozsef Kövecses
Professor, Department of Mechanical Engineering
McGill University

OCIMAE Members: Eric Lanteigne
Associate Professor, Department of Mechanical Engineering
University of Ottawa

Jeremy Laliberté
Professor, Department of Mechanical and Aerospace Engineering
Carleton University

Roland Bouffanais
Associate Professor, Department of Mechanical Engineering
University of Ottawa

Supervisors: Davide Spinello
Professor, Department of Mechanical Engineering
University of Ottawa

Mohammed Abouheaf
Associate Professor, School of Engineering
Bowling Green State University

Declaration of Authorship

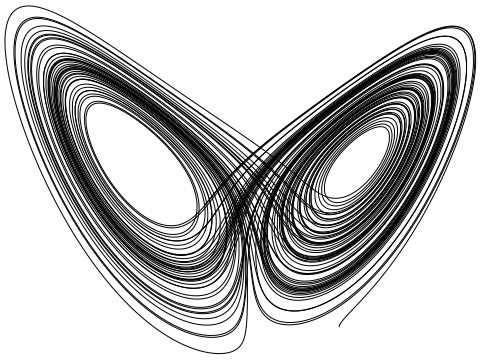
I hereby certify that this thesis is entirely my own original work except where otherwise indicated. I am aware of the University of Ottawa regulations concerning plagiarism, including those regarding consequent disciplinary actions. Any use of the works of any other author, in any form, is properly acknowledged at their point of use.

Abstract

Weight-shift controlled flexible wing ultralight aircraft are renowned for their maneuverability, low-cost operation, and high flight efficiency; yet their potential for broader applications remains untapped. No current flight control systems exist to replace the intensive manual effort and specialized training required for operating this class of aircraft. This developmental gap is attributed to modelling and control challenges presented by the complex and highly nonlinear dynamic interactions from attitude adjustment by weight-shifting, the aeroelastic properties of the flexible wing, and their resulting effects on flight behaviour. These dynamics constrain the applicability of model-based control approaches due to inherent parameter uncertainty, thereby necessitating an adaptive approach to flight control system design. This work presents novel robotic piloting systems that address the automatic weight-shift flight control problem. Fly-by-wire and computer-controlled operation is realized through the implementation of cable-driven parallel manipulators which replicate human-powered weight-shift action. Optimal coordination of the parallel actuators is achieved through a model-free approach, utilizing online adaptive critics reinforcement learning, enabling automatic piloting systems to track pitch-roll trajectories despite uncertain or unknown system dynamics and external disturbances. Through industry partner collaboration, the piloting systems are validated with real-world hardware surrogates, a flight dynamics simulator environment, and optionally-manned experimental flight test. The significant contributions of this thesis include the development of implementation techniques for online actor-critic based adaptive learning suitable for performance-constrained hardware, the design of model-free force control for cable-driven parallel robots, the creation of an open-source weight-shift controlled dynamics simulation model validated with real flight data, and the advancement of fly-by-wire weight-shift flight control technology. The results push the boundaries of weight-shift aircraft flight control technology by providing a means to enhance human pilot safety with automatic control, and pave the way for the development of unmanned and autonomous weight-shift aircraft.

"The practical, physical, and sometimes dangerous consequences of control must be respected, and the underlying principles must be clearly and well taught."

— *Dr. Gunter Stein, Respect the Unstable*



Acknowledgements

I wish to express my sincere gratitude to my long-time supervisor, Davide Spinello. Your mentorship has led me to embrace my passion for the complex, unstable, and chaotic realms of engineering. Thank you for your guidance, it is the optimum of wit and wisdom.

I am deeply grateful to my co-supervisor Mohammed Abouheaf, who helped me to bridge the gulf between the advanced concepts of adaptive control theory and my practical inclinations. Your infinite patience has been invaluable, my friend.

To Anthony Pizarro and Bruno Doerwald, the founders of my industry partner, Romaeris Corporation, thank you, your backing has been the cornerstone of this project

To my thesis proposal committee members, Eric Lanteigne and Jeremy Laliberté, thank you for your steady support over many years.

I extend my thanks to Wail Gueaieb, who first invited me to this opportunity and motivated me from the start to take my research seriously.

To the maintainers, developers, and greater community behind the ArduPilot project, your work has been critical to my success. I could not have achieved this without you.

Finally, I would like to say thank you to my long-standing colleagues, Riyanson, Luc, Teresa, Gilmar, Trevor, Boyan, Jaye, Vlad, and Ross, for their extensive assistance and invaluable insights throughout the years.

I sincerely thank Mitacs and the Natural Sciences and Engineering Research Council of Canada (NSERC) for their financial support, which has been instrumental in the completion of this project. This support has enabled me to pursue my academic career. Their commitment to fostering innovation and excellence in research is a significant benefit to Canadian society.

Dedication

Kimberley & Francis, for my foundation,

My long-time friends, for the welcome distractions,

Bella, for bringing me unwavering joy,

Hailey, for your enduring love,

Thank you.

Preface

This manuscript is structured as a thesis by papers. All papers have been submitted to Institute of Electrical and Electronics Engineers (IEEE) publications that require formal peer-review prior to offering acceptance for publication. In accordance with the University of Ottawa's academic regulations article *C-7, Thesis*, the sub-article *C-7.3, Thesis Formats* specifies the requirements for the thesis-by-articles format. The following requirements determined the necessary contents of the thesis:

a) A preface that specifies any approvals obtained to conduct the research, clearly identifies the student's contribution, and distinguishes the latter from those of collaborators, co-authors or other researchers, if any.
b) An abstract.
c) A general introduction which presents the subject of the thesis and explains relevant aspects of the subject treated.
d) The main body of the text, which may be divided into sections.
e) A conclusion.
f) A bibliography.
g) All approvals required to conduct the research and/or write the thesis: e.g., ethical approval, copyright approval, etc.

I declare that I am the principal contributor for all papers comprising the project, as specified in the following section titled *Contributions of Co-Authors*. All work contained within the thesis was prepared during my period of enrolment in the program of study, and will not appear in any other student's thesis document.

To produce a cohesive document, this thesis incorporates modified versions of the original papers. Although the core content remains largely unchanged, the format has been revised for consistency, standardizing the conventions for sections, figures, tables, equations, and algorithms. A consistent order of cited works is present across chapters. Chapter 2 presents a comprehensive literature review that unifies and substantially expands upon the literature references that are presented within the original introductions for each paper. Chapters 3 to 7 are based upon the primary content of each paper. The papers in their original format are included in Appendix A for reference.

Contributions of Co-Authors

The papers contained within the chapters that constitute the main body of the thesis are co-authored works. I served as the principal contributor, coordinating the work and overseeing the research program that underpins the thesis.

Chapter 3: An Online Reinforcement Learning Wing-Tracking Mechanism for Flexible Wing Aircraft [1]

This chapter is based upon the paper "An Online Reinforcement Learning Wing-Tracking Mechanism for Flexible Wing Aircraft", that was published in the proceedings of the 2019 IEEE International Symposium of Robotic and Sensors Environments (ROSE), held in Ottawa, ON, Canada, DOI: 10.1109/ROSE.2019.8790425. In addition, I presented the paper during a regular session in the conference.

This work was performed alongside Dr. Mohammed Abouheaf, under the supervision of Dr. Wail Gueaieb, who are co-authors of the paper. Dr. Mohammed Abouheaf and I are co-first authors. I provided the kinematic modelling for the candidate parallel actuation design that is the central problem of the paper, performed the experiments used to gather kinematic data from a real prototype of the candidate design, implemented the forward kinematics model within the simulations, introduced the idea to use the PID-like model-free structure, and conceived of the actuator mixing heuristic logic found during the experimental work. Dr. Abouheaf provided the mathematical framework for the adaptive control action, and implemented the simulation for the adaptive flight control applied to the developed kinematic model. Dr. Abouheaf and I drafted the paper collaboratively. Dr. Gueaieb provided editing and technical support.

Chapter 4: Guidance Mechanism for Flexible Wing Aircraft Using Measurement Interfaced-Machine Learning Platform [2]

This chapter is based upon the paper "Guidance Mechanism for Flexible Wing Aircraft Using Measurement-Interfaced Machine Learning Platform", that was published in IEEE Transactions on Instrumentation and Measurement, 69(7):4637-4648, July 2020,
DOI: 10.1109/TIM.2020.2985553.

This work was performed alongside Dr. Mohammed Abouheaf, under the supervision of Dr. Wail Gueaieb and Dr. Davide Spinello, who are co-authors of the paper. Dr. Mohammed Abouheaf and I are co-first authors. I designed the experimental instrumentation and

measurement platform, including the mock-up mechanical system that is analogous to the pitch kinematics of the weight-shift aircraft, implemented the online reinforcement learning control algorithm directly with the flight controller software, performed the experiments, acquired the data and performed the analysis and interpretation of the results. I designed the dual actor-critic tracking and stabilizing support systems, and introduced the idea of the guided search technique. Dr. Abouheaf provided the mathematical framework for the adaptive control action. Dr. Abouheaf and I drafted the paper collaboratively. Dr. Spinello aided in the theoretical development of the approach used in the implementation, and alongside Dr. Gueaieb provided editing and technical support.

Chapter 5: uWSC Aircraft Simulator: A Gazebo-Based Model for Uncrewed Weight-Shift Control Aircraft Flight Simulation [3]

This chapter is based upon the paper "uWSC Aircraft Simulator: A Gazebo-Based Model for Uncrewed Weight-Shift Control Aircraft Flight Simulation", that was published in the proceedings of the 2023 IEEE International Symposium of Robotic and Sensors Environments (ROSE), held in Tokyo, Japan, DOI: 10.1109/ROSE60297.2023. In addition, I presented the paper during a regular session in the conference, and the developed simulation model was contributed to an official open-source repository.

This work was a collaboration with Mrs. Teresa de Jesus Krings, Mr. Boyan Zhou, and Mr. Gilmar Tuta Navajas, and was performed under the supervision of Dr. Davide Spinello; all of whom are co-authors of the paper. I was the lead on the work, responsible for the overall conception, design, and implementation. I developed the concept for the initial dynamical model implementation based on existing theory from literature, along with my own past experience analyzing data collected from weight-shift aircraft flights. Mr. Tuta Navajas and I developed the preliminary simulation modelling implementation that served as the basis for further developments. Mr. Zhou and I developed the aerodynamics modelling approach, and the implementation of the simulated software in the loop flight controller. Mrs. Krings and I developed the process for flight data collection, performed simulated flight test, and performed the analysis of the collected data. I contributed the interpretation of the data, and drafted the paper, and contributed the model to an open-source repository. Dr. Spinello provided editing and technical support.

Chapter 6: Model-Free Force Control of Cable-Driven Parallel Manipulators for Weight-Shift Aircraft Actuation [4]

This chapter is based upon the paper "Model-Free Force Control of Cable-Driven Parallel Manipulators for Weight-Shift Aircraft Actuation", that was published in IEEE Transactions on Instrumentation and Measurement, 73:1-8, Art no. 2505108, December 2023, DOI: 10.1109/TIM.2023.3346524.

This work was performed under the supervision of Dr. Davide Spinello and Dr. Mohammed Abouheaf, who are co-authors of the paper. I am the sole first author. I designed the control problem solution, implemented the solution with the instrumentation and measurement system, designed the experimental mock-up used to acquire the data, and performed analysis of the data, and drafted the paper. Dr. Abouheaf and Dr. Spinello aided in the theoretical development of the approach, and provided editing and technical support.

Chapter 7: Automatic Weight-Shift Aircraft Control with Experimental Fly-By-Wire Flights [5]

This chapter is based upon the paper "Automatic Weight-Shift Aircraft Control with Experimental Fly-By-Wire Flights," that was submitted to the 2025 CCToMM Symposium on Mechanisms, Machines, and Mechatronics (M3 Symposium), to be held in Ottawa, Ontario, Canada, June 19-20, 2025.

This work was a collaboration with Mrs. Teresa de Jesus Krings, and was performed under the supervision of Dr. Davide Spinello; both of whom are co-authors of the paper. I was the lead on the work, responsible for the overall conception and design of the experimental fly-by-wire control system used in this study. I worked alongside the engineering team and flight operations team of the industry partner to implement the piloting system used to acquire the experimental data, performed the data analysis, and drafted the manuscript. Mrs. Krings contributed to the flight data analysis and interpretation of results. Dr. Spinello aided in the theoretical development of the approach, and provided editing and technical support.

Table of Contents

List of Tables **xviii**

List of Figures **xix**

Glossary **xxv**

Abbreviations **xxx**

1 Introduction **1**

1.1 Motivation & context 1

1.2 Considerations for engineering research ethics 5

1.3 Research objectives & corresponding contributions 8

1.4 Overarching methodology 11

1.5 Thesis outline 13

2 Literature Review **16**

2.1 Foreword 16

2.2 Model-based and adaptive model-based control 18
2.3 Model-free control 20
2.4 Optimal control by ADP techniques 23
2.5 Modelling dynamics and control of WSC flexible wing aircraft 27
2.6 Overview of CDPRs design, dynamics, and control 34
2.7 Open-source mobile robotics simulation 37
2.8 Summary of literature review 41

3 An Online Reinforcement Learning Wing-Tracking Mechanism for Flexible Wing Aircraft [1] 43

3.1 Foreword 43
3.2 Weight-shift manipulator model 45
3.3 Controller design 47
3.4 The optimal control problem 49
3.5 Neural network solution mechanism 52
3.6 Simulation results 53
3.7 Chapter summary 58

4 Guidance Mechanism for Flexible Wing Aircraft Using Measurement-Interfaced Machine Learning Platform [2] 59

4.1 Foreword 59
4.2 Operation of flexible-wing aircraft 62
4.3 Instrumentation and measurement platform 64

4.4 The control and optimization problem 65
4.4.1 Optimal control structure 65
4.4.2 Mathematical solution framework 67
4.5 Machine learning platform 69
4.5.1 Guided search process 69
4.5.2 Value iteration algorithm 70
4.5.3 Adaptive critics implementation 71
4.5.4 Scalability of the learning mechanism 74
4.6 Experimental results 75
4.6.1 Learning parameters 75
4.6.2 Test scenarios 76
4.7 Chapter summary 82

5 uWSC Aircraft Simulator: A Gazebo-Based Model For Uncrewed Weight-Shift Control Aircraft Flight Simulation [3] 83

5.1 Foreword 83
5.2 Weight-shift aircraft Gazebo model 85
5.3 Flight benchmarking results 92
5.4 Chapter summary 97

6 Model-Free Force Control of Cable-Driven Parallel Manipulators for Weight-Shift Aircraft Actuation [4] 98

6.1 Foreword 98

6.2 Weight-shift flight control mechanism 100
6.2.1 Schematic of the wing-fuselage piloting system 101
6.2.2 Description of the cable-driven parallel robot 102
6.3 Online model-free optimal trajectory tracking 103
6.3.1 Optimal trajectory tracking scheme by online model-free ADHDP . 104
6.3.2 Actor-critic solution for online value function approximation 105
6.3.3 Implementation of weight-shift control on the CDPR with multi-agent learning 107
6.4 Experimental results and discussion 108
6.4.1 Real-time learning control with weight-shift mechanism 109
6.4.2 Tracking of weight-shift s-turn commands 112
6.4.3 Summary of experimental results 114
6.5 Adaptive flight control simulation 114
6.5.1 Modified actor-critic algorithm implementation 115
6.5.2 Experimental result using two-body weight-shift mechanism surrogate with CDPR system 116
6.5.3 Adaptive weight-shift flight control result with uWSC aircraft simulator 119
6.6 Chapter summary 124

7 Automatic Weight-Shift Aircraft Control with Experimental Fly-By-Wire Flights [5] 125

7.1 Foreword 125
7.2 Federal regulations on ultralight flight 126

7.2.1 Canada 126
7.2.2 United States of America 128
7.2.3 Mexico 128
7.3 Experimental flight test 129
7.3.1 Flight control hardware and instrumentation 129
7.3.2 Experimental weight-shift aircraft 131
7.4 Experimental results 131
7.4.1 Sample 1: Maiden test of CDPR piloting system 131
7.4.2 Sample 2: Automatic flight control with CDPR 134
7.5 Chapter summary 137

8 Conclusion **138**
8.1 Summary of thesis work 138
8.2 Ongoing and future R&D directions 140
8.2.1 Data-driven approaches to learn the dynamics of parallel robots . . 140
8.2.2 Realizing new unmanned weight-shift aircraft prototypes 144

References **147**

APPENDIX **172**

A Copies of the Papers in Original Format **172**
A.1 An Online Reinforcement Learning Wing-Tracking Mechanism for Flexible Wing Aircraft 173

A.2 Guidance Mechanism for Flexible Wing Aircraft Using Measurement-Interfaced Machine Learning Platform 181

A.3 uWSC Aircraft Simulator: A Gazebo-Based Model for Uncrewed Weight-Shift Control Aircraft Flight Simulation 195

A.4 Model-Free Force Control of Cable-Driven Parallel Manipulators for Weight-Shift Aircraft Actuation 204

A.5 Automatic Weight-Shift Aircraft Control with Experimental Fly-By-Wire Flights 213

List of Tables

1.1 The methodological primitives, and key findings corresponding to each chapter of the thesis. 12

2.1 Summary of key literature references. 42

5.1 The Unified Robot Description Format (URDF) body structure name field entries. 87

5.3 Mass, inertia tensor, and relative center of mass location for primary components of the uWSC aircraft Gazebo model 87

5.4 Aerodynamic parameters pertaining to wing root and wing tip sections of the uWSC aircraft Gazebo model 91

5.5 Aerodynamic parameters pertaining to propeller airfoil of the uWSC aircraft Gazebo model 92

5.6 Comparison of key flight dynamics mean and standard deviation for the real and simulated flight trajectories over all thirteen provided samples. 94

6.1 Torque division logic, marks indicate the servos activated for the corresponding attitude correction. 108

6.2 Learning controller experimental parameters. 111

List of Figures

1.1 A prototype unmanned weight-shift control aircraft, with a cable-driven parallel manipulator system attached. WSC aircraft consist of two main bodies: the wing and the fuselage. 2

1.2 Schematic of a flexible wing weight-shift aircraft highlighting key mechanical subsystems. The four servo-winches on the fuselage, and the cable attachment points on the control bar are shown. 3

1.3 Free body diagram sequence shows the dynamics of the weight-shift aircraft during a banking roll. (a) The pilot induces dynamic imbalance by shifting weight, (b) in the global reference frame where gravity acts downward the wing orientation changes as a result of the weight-shift effect, (c) after a few moments, the dynamic equilibrium is reestablished, resulting in a banking turn. 6

1.4 Free body diagram sequence for the adverse yaw transient. (a) Weight-shift roll causes billow shift, resulting lateral lift gradient (b) adverse yaw occurs, relative wind induces corrective aerodynamic moment, (c) yaw passively corrects and aircraft enters the desired bank. 7

3.1 CAD model of parallel rigid linear actuator concept considered by the industry partner for weight-shift control. 45

3.2 Schematic of the wing-fuselage system, connected by a hang-block joint with 2 DoF, ϕ pitch and θ roll, that links the wing-keel to the mast that itself is rigidly attached to the fuselage (chassis). The two attached to the mast at P linear actuators push the wing via the control bar. 46

3.3 Coupled Pitch-Roll Control Mechanism with Wing-Fuselage Roll Disturbances. 47

3.4 The tracking control unit, time delayed error signals are used for PID-like input data structure. 50

3.5 Longitudinal motion. (a) The adaptation of the controller's gains. (b) The actuation lengths in response to desired maneuver. (c) The resulting wing orientation. 55

3.6 Lateral motion. (a) The adaptation of the controller's gains. (b) The actuation lengths in response to desired maneuver. (c) The resulting wing orientation. 56

3.7 The unadjusted actuation lengths by the different controllers. 57

3.8 Dynamic response of the coupled maneuver. (a) The actuation lengths. (b) The coupled longitudinal-lateral maneuvers. 57

4.1 Experimental mock-up used to emulate the motion of the fuselage about the wing in ground test. Note that the wing-mast system is oriented upside down; i.e., the top surface of the middle bar, representing the wing, is facing the floor, while the mast likewise sits above it. The figure also shows the Navio2+Raspi IMU and controller, servomotor winches, and hang-block. . 60

4.2 Schematic of weight shift kinematics of flexible-wing aircraft with two servo winches acting on wing to affect pitch. 63

4.3 Flow schematic of experimental instrumentation. 65

4.4 Feedback control loop of pitch weight-shift control mechanism. The adaptive error tracking unit and stability support unit cooperate to achieve the trajectory tracking objective. 74

4.5 Control performance during online learning process: (a) measured vs. desired pitch attitude, (b) acting forces on the wing's keel, f_{fore} and f_{aft}. . . 79

4.6 Evolution of actor-critic weights during learning process: (a) tracking critic unit, (b) tracking actor unit, (c) stabilizing critic unit, (d) stabilizing actor unit. 79

4.7 Evolution of the policy evaluation metric. 80

4.8 Control performance while learning with a converged controller in the presence of mechanical disturbances: (a) measured vs. desired pitch attitude, (b) acting forces on the wing's keel, f_{fore} and f_{aft}. 80

4.9 Variations in critic weights in the face of mechanical disturbances: (a) tracking unit, (b) stabilizing unit. 81

4.10 Control performance with previously learned static control policies: (a) measured vs. desired pitch attitude, (b) acting forces on the wing's keel, f_{fore} and f_{aft}. 81

5.1 Capture from Gazebo simulator of the uWSC aircraft model performing a starboard side banking turn. The hung mass under the wing is responsible for attitude adjustments via weight-shifting control. 85

5.2 3D CAD model of the Romaeris uncrewed weight-shift aircraft prototype. . 86

5.3 Gazebo model visualized with primary components labelled. 88

5.4 Tree diagram representation of the uWSC aircraft model as defined in the standard definition format file that is used by ROS. Black boxes are rigid bodies defined as a combination of a link and a 3D collision volume, Arrows represent injective rigidity maps, XYZ and RPY are pose vectors specify the translation and rotation of each link relative to its rigid parent, and blue ellipses are 1 DoF joint functions. See acronyms definitions in 5.1. 89

5.5 (a) Projected ground track and (b) altitude over time for a single mission, comparing simulated and real flight data. 95

5.6 Flight trajectories according to four flight phase subcategories, comparing the simulated and real flight data. 96

6.1 Free body diagram sequence illustrating the dynamics of the weight-shift aircraft control during a banking roll: (a) Beginning in balance, (b) pilot induced imbalance produces a weight-shifted dynamic equilibrium, (c) resulting in a banking turn when dynamic balance is restored. 101

6.2 Schematic (a) of wing-fuselage system with attached CDPR piloting system; (b) the corresponding kinematic diagram. 102

6.3 Trajectory stabilizing actor-critic model-free controller. 107

6.4 Multi-agent learning controller for the cable-driven parallel robot. The agents share the set of servomotors to actuate the weight-shift mechanism. 109

6.5 Experimental weight-shift control mock-up used to emulate the weight-shift effect of the wing controlled by the CDPR. The wing keel and control bar (blue) are suspended at the hang-block, and are rotated by weight-shift of the fuselage (yellow) by forces on the parallel cables (green). 110

6.6 Trajectory tracking of wing's pitch and roll for coordinated S-turns. 112

6.7 Normalized servo torque commands. 113

6.8 Trajectory tracking with sustained disturbance during learning, beginning at 10 s. 113

6.9 Tracking performance during learning. The tracking error decreases as learning progresses. 117

6.10 (a) Actor neural network weights. (b) Critic neural network weights. 117

6.11 (a) Accumulation of value over time. (b) Temporal difference error evolution during training. (c) Temporal difference error, with 100 step and 1500 step moving averages. (d) Critic NN weights, log weighted with 100 step moving average. 118

6.12 Initial training for ground pre-tune of actuation control mechanism in the roll input, (a) reference track (b) resulting actor weights. 119

6.13 Initial training for ground pre-tune of actuation control mechanism in the pitch input, (a) reference track (b) resulting actor weights. 120

6.14 Simulated uWSC flight employing the adaptive actor-critic implementation for flight control. 122

6.15 Adaptive optimal trajectory tracking roll control of weight-shift aircraft via online reinforcement learning, (a) reference track (b) resulting actor weights. 122

6.16 Adaptive optimal pitch trajectory tracking control of weight-shift aircraft via online reinforcement learning, (a) reference track (b) resulting actor weights. 123

7.1 Examples of lab experiment hardware employed to test the CDPR piloting system prior to flight test. 130

7.2 The CDPR piloting system equipped to weight-shift aircraft. When engaged, the CDPR replaces physical pilot control with electronic FBW control provided by the controller. 132

7.3 Flight trajectory of first sample, with piloting system inputs for coordinated turn maneuver. 133

7.4 Flight trajectory of the second sample. The automatic flight controls were active for the track sections labelled FBW-1 and FBW-2. 134

7.5 Roll controller input and response of CDPR piloting system while in FBW-1 section of test flight sample. 135

7.6 Roll controller input and response of the CDPR piloting system while in FBW-1 for the large sustained clockwise turn. 136

8.1 Results from DMDc where linear dynamic model is trained with data captured from CDPR. 142

8.2 Results from DMDc test using trained model against validation data. . . . 143

8.3 Example mission scenarios considered using the simulator methodology. (a) The uWSC model detecting gas emissions. (b) A multi-agent approach for the emissions detection mission. 144

8.4 Simulation and maiden flight images of the prototype unmanned weight-shift aircraft, showcasing the progression from Gazebo modelling to real-world testing. The top image displays the Gazebo model, while the bottom series illustrates a sequence in the first flight, demonstrating the practical application of the design and control developed in the simulation. 145

8.5 First autonomous uWSC flight, circa November 2024, (a) Waypoint navigation track, (b) An image of the aircraft flying overhead. 146

Glossary

Actor-Critic An on-policy reinforcement learning algorithm where one part (actor) decides actions, and another (critic) evaluates them

Actuator A device that moves or controls parts of a system

Adaptive Control A control method that adjusts itself according to changing operating conditions

Adaptive Critics An on-policy reinforcement learning framework that evaluates and iteratively improve control policies

Aeroelastic The interaction between aerodynamic forces and structural elasticity

Approximate Dynamic Programming A body of techniques that combine reinforcement learning and dynamic programming to solve optimal control problems where exact solutions are impractical due to computational limits or model uncertainties

ArduPilot An open-source autopilot software widely used for controlling unmanned aerial vehicles and other mobile autonomous systems

Bellman Optimality Principle A rule that taking the best choice at any moment leads to the best overall result, independent of previous actions

Cable-Driven Parallel Robot A type of robotic system where the end-effector is controlled and positioned through a set of tensioned cables driven by winches.

Costate A mathematical variable used in optimal control theory, representing the sensitivity of the performance criterion to changes in the state variables, and often associated with Lagrange multipliers in the Hamiltonian formulation

Curse of Dimensionality A phenomenon where the computational and data requirements of solving high-dimensional problems, such as those involving continuous state and action spaces, grow exponentially, rendering traditional dynamic programming approaches infeasible for real-world control problems

Disturbances Unwanted external influences or changes that affect the performance or behaviour of a system

Dynamic Programming An optimization approach that simplifies complex problems by recursively breaking them into smaller sub-problems

End-Effector The device or tool attached to the end of a robotic manipulator

Exploration The process of sampling actions or states in a system to gather information

Feedback Control Using the observed difference between the desired result and the actual result to make corrections, *otherwise known as cybernetic control*

Feedforward Control Anticipating changes to make adjustments in advance to meet targets, often incorporates a system model

Flexible Wing A type of wing structure that incorporates rigid links and flexible materials, allowing for intentional deformation of airfoil to occur under aerodynamic forces

Flight Envelope Phase The specific set of operating conditions within the flight envelope of an aircraft, characterized by variables such as airspeed, altitude, angle of attack, and load factor

Fly-By-Wire An electronic flight control system that replaces manual controls with electronically mediated inputs and outputs

Fuselage The central chassis of an aircraft that houses the crew, passengers, or cargo and connects structural components, the hang-block connection point, and landing gear

Gazebo An open-source physics-based robotics simulation software that allows for testing and validation of robotic systems in virtual environments, *notably, not ROS*

Hamilton-Jacobi-Bellman Equation A fundamental equation in optimal control theory that expresses the necessary conditions for optimality of a control policy, and defines the value function, linking the current state to the cumulative cost of future actions

Hang-Block A pivot mechanism in weight-shift trike aircraft connecting the keel of a wing to the mast of the fuselage (chassis)

Heuristic A practical problem-solving strategy that employs rules of thumb or educated guesses to efficiently find solutions when exact methods are computationally infeasible

Intelligent Control A control methodology that combines techniques from artificial intelligence, such as neural networks, fuzzy logic, and reinforcement learning, to handle complex, uncertain, or adaptive systems

Kinematics The branch of mechanics that deals with the motion of objects without considering the forces or masses causing the motion

Model Knowledge The degree to which a system's behaviour is understood, ranging from fully known to partially known or completely unknown

Model-Based Control Techniques for controlling systems according to a mathematical system model

Model-Free Control Techniques for controlling systems without a mathematical system model

Multi-Agent System A system involving multiple autonomous agents that interact or cooperate to achieve individual or collective goals

Neural Network A machine learning model composed of interconnected layers of artificial neurons that approximates functions

Nonlinear Dynamics Time dependent systems where outputs are not directly proportional to inputs

Off-Policy A reinforcement learning method where the policy being learned is independent of the policy generating the actions, allowing for learning from previously collected data or alternate exploration strategies

Offline Learning A learning approach where the system is trained on a fixed dataset without real-time updates

On-Policy A reinforcement learning method where the policy being learned is used to generate the following actions, enabling the agent to adapt its behaviour during exploration

Online Learning A way of reinforcement learning where the system updates itself continuously as it receives new data

Optimal Control A control methodology for finding policies or strategies that optimizes a performance criterion, subject to system dynamics and constraints

Parallel Manipulator A mechanical system with multiple actuated links moving with coordination to control a single end-effector

Parameter Variation The shifts in a system's dynamics characteristics over time because of wear, environmental effects or other disturbances

Persistent Excitation A condition where the input signals sufficiently excite all modes of the system, often a requirement for adaptation

Pitch The rotation of an aircraft around its lateral axis.

Policy A function that determines the actions an agent or system should take in response to a given state

Policy Iteration A method used to calculate the optimal policy by alternating between policy evaluation and policy improvement steps

Pre-Training An initial training phase where a model is trained on a related dataset or task to improve performance or reduce training time on the main task

Q-Learning An off-policy reinforcement learning algorithm that learns the value of possible actions given the present state

Reference Input The desired target or goal that a system aims to achieve

Reinforcement Learning A method where a system adapts by trying actions, evaluating the results, and improving over time accordingly

Reward A mathematical function that assigns a value representing the benefit or cost of a particular action or state according to designer-defined objectives, *commonly known as Utility*

Robot Operating System An open-source middleware framework for robot software development, *notably, not Gazebo*

Robust Control A control methodology designed to maintain performance despite model uncertainties and disturbances

Roll The rotation of an aircraft around its longitudinal axis.

Stability Property of a system's sensitivity to return to steady-state motion after being disturbed

State-Action-Reward-State-Action An on-policy reinforcement learning algorithm that updates the value of a state-action pair based on the previous action taken

Temporal Difference Learning A reinforcement learning method that estimates the value function by observing changes in successive value estimates

Trajectory Tracking A control where the objective is to ensure a dynamic system follows a desired time-varying reference with minimal deviation

Value Function A mathematical function that quantifies the expected cumulative reward of being in a specific state

Weight-Shift Control Aircraft Uses a mass under the wing to do flight control, think hang-gliders or air-trikes

Yaw The rotation of an aircraft around its vertical axis.

Abbreviations

ADP Approximate Dynamic Programming

CDPR Cable-Driven Parallel Robot

CoG Centre of Gravity

DMD Dynamic Mode Decomposition

DoF Degree of Freedom

HJB Hamilton-Jacobi-Bellman

OS Open Source

PID Proportional-Integral-Derivative

RL Reinforcement Learning

RPAS Remotely Piloted Aircraft System

SITL Software-In-The-Loop

TD Temporal Difference

UAS Unmanned Aircraft System

UWSC Unmanned Weight-Shift Control

VFA Value Function Approximation

WSC Weight-Shift Control

Chapter 1

Introduction

1.1 Motivation & context

Weight-shift controlled (WSC) flexible wing ultralights are a class of aircraft distinct from conventional fixed-wing flight vehicles. As opposed to conventional aerodynamic control surfaces, these aircraft maneuver by manipulating a mass attached to wings with lightweight, semi-rigid frame structure [6–8]. This class of aircraft, which includes hang-gliders and powered air trikes, are distinct from parafoil-based aircraft like paragliders. The latter lack a rigid wing structure and primarily use brake lines for flight control. WSC aircraft are appealing as a flight platform due to their simple mechanical structure, high maneuverability at low speed, self-stabilizing wing aerodynamics, ability to operate with little infrastructure on rugged terrain for take-off and landing, low cost of fabrication, and high payload-to-mass ratio. The potential mission scope of WSC aircraft is constrained by the requirement for human-powered pilot control, limiting the free payload capacity and total flight duration, while requiring a high standard for pilot safety. WSC aircraft have been relegated primarily to recreational use for enthusiast pilots, yet have seen niche use for scientific applications [9, 10]. By integrating an automatic flight control piloting system, these aircraft would be utilizable in optionally-manned and unmanned operations as a remotely piloted aircraft system (RPAS), thereby providing new opportunities and applications of the technology in missions that leverage the operational characteristics of WSC aircraft. The overarching motivation of this research pursuit is the development of automatic weight-shift flight control that matches the capability of a human pilot. The

thesis work makes significant developments in automatic weight-shift flight control, advancing the state of the art. An example of a successfully implemented prototype of the mechanical engineering solution is shown in Figure 1.1, and the corresponding schematic diagram is illustrated in Figure 1.2.

Figure 1.1: A prototype unmanned weight-shift control aircraft, with a cable-driven parallel manipulator system attached. WSC aircraft consist of two main bodies: the wing and the fuselage.

The thesis is shaped by a research fellowship with an industry partner, Romaeris Corporation, whose efforts focus on the development of unmanned WSC aircraft technology, including automatic flight controllers [11–13]. The industry partner is a purpose founded research and development (R&D) start-up who is investigating the commercialization of autonomous WSC ultralight aircraft for use as unmanned aircraft systems (UAS). The industry partner is a key collaborator within several government-industry and academic-industry projects that have contributed publicly to the advancement of technological and regulatory policy research for RPAS on several pertinent topics including beyond-visual-line-of-sight telemetry, automated detect and avoid deconfliction mechanisms, and UAS operations in protected airspace [14–17]. The scope of this thesis focuses on the exploration and development of low-level automatic flight control, robotics, and flight actuation mechanisms for WSC aircraft, with the aim of enabling higher-level flight control capa-

bilities within the broader research environment for UAS systems mentioned earlier. It is

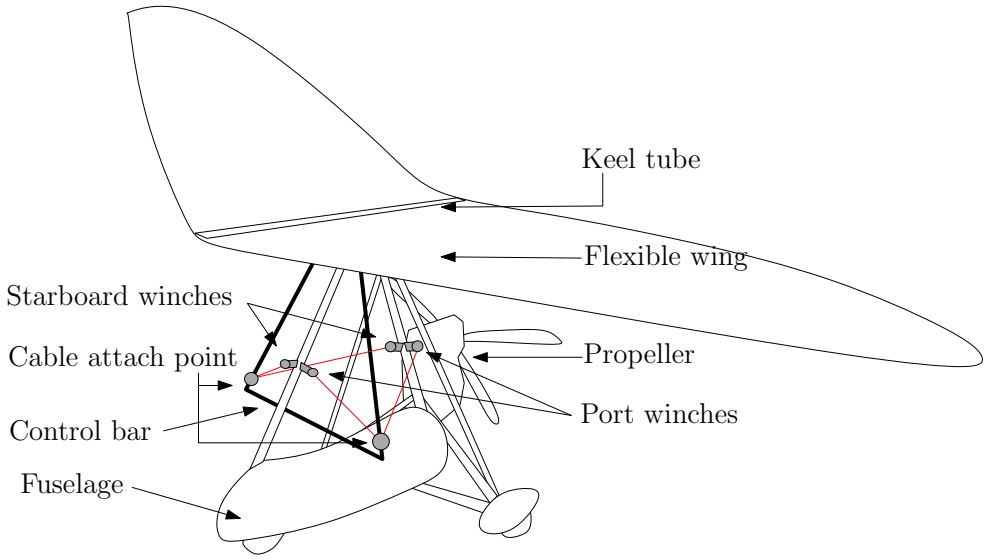


Figure 1.2: Schematic of a flexible wing weight-shift aircraft highlighting key mechanical subsystems. The four servo-winches on the fuselage, and the cable attachment points on the control bar are shown.

important to establish the foundational context for the research project. After an extensive literature review and patent search, no existing development of solutions for automatic weight-shift flight control were discovered. This fact was first identified by preliminary research discovery efforts of the industry partner and has been exhaustively validated in the course of the thesis work. Developing accurate dynamic models for flexible wing weight-shift control aircraft is challenging due to their highly nonlinear dynamics, which stem from the wing's aeroelastic properties and the two-body weight-shift relationship inherent to their flight control [18–20]. Initial development efforts made by the industry partner centred on the development of flight control via active airfoil deforming actuation schemes that take advantage of the aeroelastic properties of the flexible wing [21]. The control and dynamics modelling efforts proved to be extremely challenging as they had to account for the multi-physics of the elastic properties of the wing structure along with the corresponding aerodynamics of the variable airfoil shape that depend strongly on the aircraft's state. Despite some success for the proof of concept in experimental flight tests, the approach was ultimately deemed infeasible due to the low flight control authority it provided as the sole automatic flight control method. The industry partner further considered alternative approaches with the addition of conventional flight control surfaces as an avenue for

automatic flight control. After preliminary flight test experiments, the industry partner concluded that flight control surfaces could not provide the basic flight behaviour that standard human piloted WSC allows for. Around the same time, Abouheaf et al. circumvented the nonlinear flight dynamics modelling concerns by demonstrating stabilizing weight-shift controls for a two-body hang-glider model developed with adaptive model-free control approaches including reinforcement learning (RL) [22–27]. These results led the industry partner to explore the feasibility of attitude control provided by robotic actuation methods that would mimic a human pilot. Unlike previous flight control methods studied by the industry partner, this hypothetical piloting system would apply moments to the wing by weight-shift reactions induced by moving the suspended mass.

Dynamic weight balancing is the central principle underlying WSC aircraft maneuvering. The thesis places a strong emphasis on the tenet of understanding manned WSC flight, as this was developed through extensive consultation with WSC pilots that worked with the industry partner [8, 28]. Weight-shift flight control requires that the pilot enacts the correct force input on the control bar of the wing dependent on the state of the vehicle relative to its flight envelope. The pilot manipulates the motion of the weight attached to the flexible wing to perform banking turns or speed-changing pitch maneuvers without conventional aerodynamic control surfaces. The pilot manipulates the dynamic balance for the aircraft by applying force against the control bar, inducing intentional dynamic imbalance. When applying force input on the control bar, the reaction moves the fuselage body, relative to the wing body, modifying the centre of gravity (CoG) for the total aircraft. The fuselage motion, as well as some small motion of the wing due to the initial reaction to the force input, causes a dynamic imbalance as the centre of gravity of the two bodies become misaligned with respect to gravity. An apparent body moment acts on the wing until this balance is restored, causing a change in the lift vector direction. The example of a banking turn is illustrated in Figure 1.3. Pitch control is performed in a similar fashion along the longitudinal axis. Notably, the transient motion before restoring equilibrium between the wing and fuselage during weight-shift is non-minimum phase for both the lateral and longitudinal dynamics [18, 29, 30].

With non-minimum phase systems, the initial controller input must cause an undershoot that makes it seem like the system is responding in the opposite direction than the intended target, temporarily increasing error [31]. After some time, the system's phase trajectory recovers and converges towards the target. As an inherent property of these systems, stabilizing controls are limited to a bandwidth range that demands trade-offs between responsiveness, stability, and sensitivity that may not be feasible [32].

Consider the lateral steering dynamics of single track systems like bicycles or motorcycles, which are non-minimum phase [33]. Novices often struggle with steering because the initial control input requires turning the handlebars briefly in the opposite direction of the intended turn. This counter-steering action initiates a lean into the turn, inducing the bike to enter a dynamic non-equilibrium transient before stabilizing. Eventually, the rider masters this unintuitive dynamic of steering control. Similarly, in weight-shift aircraft, the initial response to pilot input in both the lateral and longitudinal axes is also non-minimum phase, presenting analogous challenges.

For a banking turn control input, the correct weight-shift action initially induces an adverse yaw opposite to the desired heading due to the wing twisting along its lateral axis. The deformation modifies the angle of attack for each half, resulting in increased lift on one half of the wing and decreased lift on the other, an effect known as billow-shift. It is only after a few moments of applying the correct coordinated weight-shift input that the wing enters the intended banking turn and restores dynamic equilibrium. Figure 1.4 illustrates the transient behaviour that occurs when transitioning from level trim flight to a banking turn. Similarly for elevation control, the correct weight-shift action initially induces the opposite of intended elevation change. At first the entire aircraft rotates about the lateral axis, causing a change in airspeed opposite to the desired climb due to the modified angle of attack and consequent drag. After a few moments, the sustained weight-shift input converges to the desired elevation rate change and dynamic equilibrium is restored. For novice weight-shift pilots, these coordinated control inputs feel completely counter-intuitive and require significant training, much like riding a bicycle for the first time. A hypothetical automatic weight-shift control piloting system must respect the higher sensitivity to disturbances while managing the complicated control dynamics well.

1.2 Considerations for engineering research ethics

Researchers on the project exercise a duty of care in that all hardware and software generated from the work is validated extensively through surrogate experiments prior to flight test. Hardware systems must be equipped with emergency power stops as well as hard-coded firmware limits in control force produced. All flight vehicle operations are solely the domain of Romaeris, and their certified pilot operators with WSC training. Flight test regulations for experimental aircraft are at the root of the operations conducted by Romaeris [8, 34–37]. Great care is taken to minimize risk to the test pilot and greater

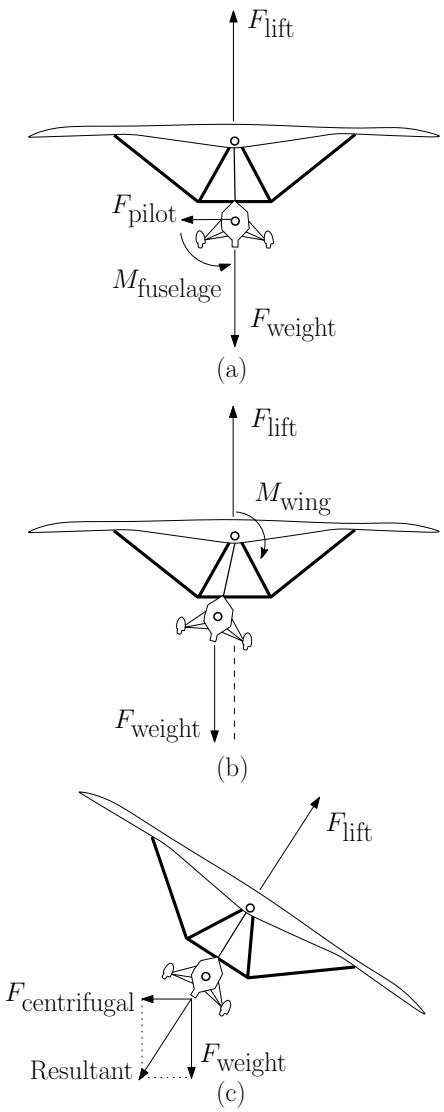


Figure 1.3: Free body diagram sequence shows the dynamics of the weight-shift aircraft during a banking roll. (a) The pilot induces dynamic imbalance by shifting weight, (b) in the global reference frame where gravity acts downward the wing orientation changes as a result of the weight-shift effect, (c) after a few moments, the dynamic equilibrium is reestablished, resulting in a banking turn.

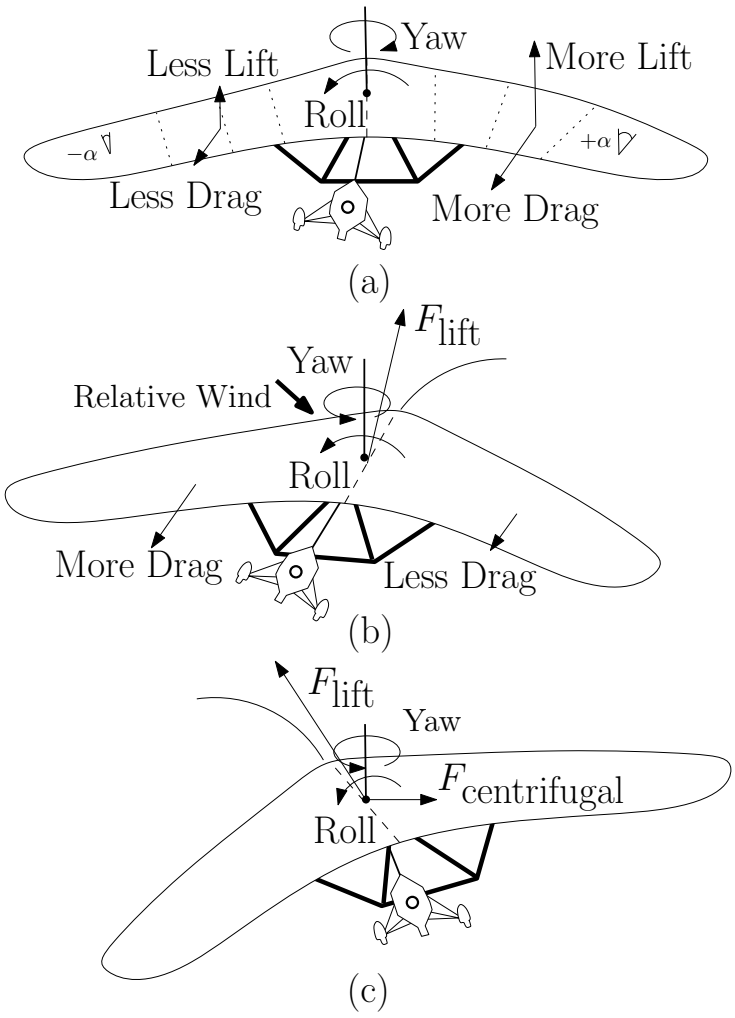


Figure 1.4: Free body diagram sequence for the adverse yaw transient. (a) Weight-shift roll causes billow shift, resulting lateral lift gradient (b) adverse yaw occurs, relative wind induces corrective aerodynamic moment, (c) yaw passively corrects and aircraft enters the desired bank.

public while following all aviation regulations in place. Chapter 7 presents a summary of federal flight regulations pertaining to ultralights in Canada, the USA, and Mexico.

The rights for the intellectual property stemming from the research is governed by a pre-existing agreement between the research team and Romaeris that outlines ownership and usage. The agreement ensures that scholarly contributions and experimental data are publishable, unimpeded by the commercialization goals of the company. Central to this agreement, all published work must allow for reproducibility and evaluation of the research quality. Results remain objective and accurate. Some reasonable limitations are placed on the sharing of non-research details of the technology that are considered as 'trade secrets' of Romaeris, hence potential research publications are reviewed by Romaeris in advance of submission.

Several of the methods selected for the research leverage open-source frameworks, hence there is an expectation that contributions will be made, returning value to the open-source communities. All open-source license agreements are followed. Software tools developed with Gazebo are governed by the Apache 2.0 license [38], while applications developed with Ardupilot are governed by the GNU GPL [39].

1.3 Research objectives & corresponding contributions

The work aims to address the research gap defined by the absence of automatic weight-shift flight control and make advancements to the state of the art. The overarching research objectives are summarized with the following logical progression:

1. Develop an understanding of basic WSC flight principles through experimental flight test analysis, expert pilot consultation, existing literature study, and approximation of nonlinear dynamics using an open-source robot simulation methodology that incorporates expert pilot input.
2. Design, develop, and validate experimental parallel actuation hardware for WSC ultralights, addressing forward and inverse kinematics of cable-driven parallel robots (CDPRs) using model-free techniques, and test the proposed solutions on real-world hardware, including experimental flight test vehicles.
3. Design and experimentally validate adaptive critics based online RL schemes that are suitable for real-time low-latency controllers with compute performance constraints.

4. Demonstrate the feasibility of automatic weight-shift flight control by the proposed solutions, making significant advancements to the state of the art.

The main contributions from the research have been validated through experimental testing of the piloting systems. All development leverages WSC pilot expertise provided by the industry partner. The control and robotic systems have been comprehensively tested with kinematic surrogate apparatuses, simulated flight scenarios, and real-world experimental flight tests. They are summarized as follows:

1. **A prototype robot piloting system concept that is experimentally validated to match the WSC capability of a human pilot**.
    a) Demonstrate feasibility of empirically derived forward kinematic model for candidate parallel manipulators to control hang-glider model (Chapter 3).
    b) Experiment with single-axis WSC actuation by cable-driven servo-winches (Chapter 4).
    c) Implement 2-axis WSC by CDPR prototype, demonstrate experimentally on test hardware (Chapter 6).
    d) Map fly-by-wire (FBW) WSC via CDPR to higher level flight control with human pilot in the loop (Chapter 6).
    e) Flight test that validate feasibility of solution (Chapter 7).
    f) Demonstrate automatic control for RPAS flight test (Chapter 8).
2. **Model-free learning techniques for trajectory tracking of a CDPR's end-effector while optimizing parallel manipulator coordination**.
    a) Demonstrate parallel actuator coordination through basic mixing heuristic (Chapter 3, 6).
    b) Employ model-free control approach to track a commanded reference trajectory for a single-axis with parallel cable-driving actuators (Chapter 4).
    c) Generalize method to enable solution to track two-axis reference trajectory (pitch-roll) simultaneously with a set of parallel cable-driven manipulators (Chapter 6).
    d) Employ multi-agent actor-critic learning to train separate pitch tracking and roll tracking units that coordinate trajectory tracking across shared set of parallel cable-driven actuators (Chapter 6).

e) Design an algorithm that learns approximations the input mapping for a parallel manipulator with unknown manipulator placement, allowing for the optimization of control effort (Chapter 8).

3. **Design of adaptive critics algorithms for implementation with computationally-constrained systems requiring low-latency in control updates**.

   a) Implement and test existing online model-free policy-iteration and value-iteration RL methods with neural networks (Chapter 4).

   b) Experiment with different control policy evaluation schemes, including least square, recursive least square, and gradient descent (Chapter 3, 4).

   c) Develop modifications to the adaptive critics scheme based on value iteration that is computationally low-cost, i.e. single step update of value approximations (Chapter 4).

   d) Validate by real test of on implementation in real-time control software on embedded flight control systems (Chapter 4, 6).

   e) Develop improvements to learning algorithm method to increase flexibility with initial admissible policy (Chapter 6).

   f) Develop actor update law with its own gradient descent rule to allow for different adaptation rates between critic and actor (Chapter 4).

4. **A WSC ultralight aircraft model that captures the corresponding approximate nonlinear flight dynamics, harnessing OS simulation methodologies**.

   a) Study existing Gazebo based simulation models and packages for software in the loop flight controller testing (Chapter 5).

   b) Develop an approximate inertial and aerodynamic model of the ultralight that captures key lateral and longitudinal stability properties (Chapter 5).

   c) Simulated flight test that demonstrate feasibility of automatic WSC through the mapping of outputs of a conventional 2-axis high-level flight controller to weight-shift manipulations, including automatic waypoint navigation controller (Chapter 5).

   d) Dissemination of the model to the open-source robotics simulator community (Contributed in [40]).

   e) Test environment for online model-free RL control schemes with the simulator ultralight model (Chapter 6).

1.4 Overarching methodology

The research work employed a staged approach to address automatic weight-shift flight control for flexible wing aircraft. The overarching methodology advanced from foundational modelling and simulation to real-world experimentation. Relevant methodologies are incorporated in each chapter, as summarized in Table 1.1. The iterative feedback loop between hardware experiments and simulations refined the overall solutions. The work culminates with a piloting system proven in experimental flight test.

First, weight-shift flight dynamics were understood using simulation models informed by expert pilot input [28] and real flight data. Detailed in Chapter 3 and Chapter 5, this approach captured the essential nonlinear dynamics of WSC aircraft. The use of open-source simulation frameworks enabled flexible experimentation, resulting in more rapid controller design iteration and the selection of computationally feasible model-free control schemes.

Next, a robotic actuation system using parallel actuators replicate human pilot weight-shift inputs is developed, as detailed in Chapters 4 and 6. The CDPRs provided a high stiffness-to-weight ratio, reconfigurable geometry, and variable compliance, making them suitable for retrofits to existing weight-shift aircraft. Adaptive controllers based on online RL were implemented on resource-limited embedded controllers and validated through surrogate mock-up experiments of weight-shifting mechanisms. Incremental validation ensured their robustness before full-scale flight trials.

Finally, the integrated system of model-free automatic controls for a CDPR was tested under real flight conditions, as shown in Chapter 7. These experiments validated the automatic weight-shift control concept, demonstrating that the adaptive methods and CDPR-based actuation approached human pilot capability. Performance was evaluated using standard flight stability observations, trajectory tracking accuracy, and pilot-defined acceptability criteria.

Table 1.1: The methodological primitives, and key findings corresponding to each chapter of the thesis.

	Chapter 3	**Chapter 4**	**Chapter 5**	**Chapter 6**	**Chapter 7**
Paper	*[1] An Online Reinforcement Learning Wing-Tracking Mechanism for Flexible Wing Aircraft*	*[2] Guidance Mechanism for Flexible Wing Aircraft Using Measurement-Interfaced Machine Learning Platform*	*[3] uWSC Aircraft Simulator: A Gazebo-Based Model for Uncrewed Weight-Shift Control Aircraft Flight Simulation*	*[4] Model-Free Force Control of Cable-Driven Parallel Manipulators for Weight-Shift Aircraft Actuation*	*[5] Automatic Weight-Shift Aircraft Control with Experimental Fly-By-Wire Flights*
Methods	Idealized hang-glider model, model-free adaptive critics, PID-like input data structures, kinematic modelling, dynamics model simulation.	Model-free adaptive critics, flight controller hardware, online RL, PID-like input data structures, surrogate robotic mock-up.	Model-free PID, ArduPilot-SITL, Gazebo physics simulation, aerodynamics modelling, validating against real flight data, surrogate mock-up.	Model-free adaptive critics, PID-like input data structures, surrogate robotic mock-up, flight controller hardware, online RL, ArduPilot SITL, Gazebo physics simulation.	Model-free PID, Experimental validation in flight, regulatory environment review, flight controller hardware.
Findings	Validated RL-based tracking control, demonstrated feasibility of stabilizing flexible wing dynamics via real-time adaptation with embedded parallel actuator kinematics.	Implementation techniques for online-RL under computational constraints, intelligent unit for stabilization support, guided search for policy updates.	uWSC Gazebo model validated against real WSC flight data, enabled safe SITL testing of control strategies before real flight.	Optimal trajectory tracking for CDPR with model-free control on input-output data, disturbance rejection property, generalizable to different actuation schemes.	Validated automatic weight-shift control with real flight test, matched human pilot capability with FBW system, advanced state-of-the-art.

1.5 Thesis outline

The remainder of the thesis is organized as follows:

Chapter 2 presents a review of the literature related to the research context to facilitate better understanding of the research work. The review unifies and substantially expands upon the literature references discussed within the original papers. First, a succinct overview of literature on model-based, model-free, adaptive, and approximate dynamic programming methods in RL, with a specific focus on their implications for flight control of WSC aircraft is presented. This provides a sense for the theoretical foundations underpinning the optimal trajectory tracking controller developments present in the research work. To help put the difficulty of the flight controller design problem in context, an exhaustive summary of the body of literature around approaches taken to model the dynamics and control for WSC flight of flexible wing aircraft is summarized. As the piloting system is realized through experimental implementations of CDPRs and with a flight simulation methodology, the remainder of the literature review details these topics. An overview of recent literature exploring the design, kinematics, and control of parallel manipulators and CDPRs is presented to give context for the robotic piloting system, and a summary is provided on the key relevant literature that inspired the open-source robotics simulation methodology. The literature review concludes with a summary that connects the referenced works to the thesis work.

Chapter 3 is adapted from the publication titled "An Online Reinforcement Learning Wing-Tracking Mechanism for Flexible Wing Aircraft". It details the application of an online RL mechanism for controlling a weight-shifting flexible wing aircraft mathematical model using a parallel actuation scheme. The model-free adaptive control integrates with the kinematic model of a candidate parallel manipulator design. The feasibility of stabilizing the aircraft's dynamics using real-time sensor feedback without requiring prior knowledge of its dynamics is examined. Simulations demonstrate the control solution's capability to adapt to dynamic changes and uncertainties in flight conditions. The findings illustrate the potential utility of this control mechanism in improving the performance and safety of weight-shift aircraft if it were implemented as a prototype piloting system.

Chapter 4 is adapted from the publication titled "Guidance Mechanism for Flexible Wing Aircraft Using Measurement-Interfaced Machine Learning Platform". It presents a model-free control for weight-shift aircraft that utilizes real-time data feedback to adapt control policy via online RL. The design and implementation of an adaptive trajectory tracking

control is realized with a pair of actor-critic learning agents that must coordinate separate tracking error correction and stability objectives. The solution is implemented directly on flight controller hardware, thus relies on several practical techniques including a learning guidance mechanism and single-step value iteration. Real-time performance feedback data is used to adapt the control strategy without depending on established dynamical models. Experimental tests validate the functionality of the solution, underlining the potential of adopting a model-free approach to the robotic actuation input of WSC aircraft.

Chapter 5 is adapted from the publication titled "uWSC Aircraft Simulator: A Gazebo-Based Model for Uncrewed Weight-Shift Control Aircraft Flight Simulation". It introduces a dynamics model for simulating the flight behaviour of unmanned weight-shift control (uWSC) aircraft. The modelling methodology integrates known mass and geometry properties of a prototype weight-shift aircraft, and flight behaviour is informed by expert weight-shift pilots. The essential static stability properties of the flexible wing is approximated by a rigid wing composed of multiple sections. The resulting aircraft model correctly approximates the interplay between aerodynamic forces and the weight-shifting mechanism that characterizes their flight performance. Validation of the model is achieved through comparative analysis with actual weight-shift aircraft flight data. The modelling approach, and resulting uWSC model present a viable method to develop flight controller software and test new scenarios, reducing the reliance on real-life flight test.

Chapter 6 is adapted from the publication titled "Model-Free Force Control of Cable-Driven Parallel Manipulators for Weight-Shift Aircraft Actuation". It details an adaptive model-free force control that uses cable-driven parallel manipulators to actuate the weight-shift mechanism. This approach utilizes online RL and does not require any model of the system. The solution is designed to operate within the computational constraints of flight controller hardware, and coordinates parallel cable tension adjustments dynamically to perform standard weight-shift flight maneuvers based on trajectory tracking performance feedback alone. Experiments are performed with a weight-shift mechanism surrogate to validate the system's capability to execute weight-shift maneuvers using the adaptive force adjustments. With recent work, the design is employed for adaptive weight-shift flight control in simulated flight test. This model-free control system presents a feasible method for automating weight-shift manipulation in future electronically controlled and unmanned weight-shift aircraft.

Chapter 7 is adapted from the paper titled "Automatic Weight-Shift Aircraft Control with Experimental Fly-By-Wire Flights". It presents the experimental application of the model-free, CDPR based weight-shift flight control piloting system in real experimental flights.

The chapter recounts the collaboration with the industry partner, emphasizing the alignment of the flight tests with established experimental flight operation protocols that are compliant with federal flight regulations. The flight test experiments involve integrating the CDPR system with a novel FBW piloting system for the flexible wing aircraft. The pilot utilizes a joystick for input to execute weight-shift flight maneuvers. The CDPR's capability to replace human-powered control is validated by analysis of the FBW flight data. The results demonstrate the piloting system's potential for controlling aircraft attitude, thus advancing the state of the art.

Chapter 8 concludes with a summary of the thesis and its contributions, along with brief discussion on directions for future research for data-driven RL techniques to optimize control strategies for robotic systems, and the development of autonomous WSC aircraft prototypes.

Appendix A includes the papers in their original format.

Chapter 2

Literature Review

2.1 Foreword

This chapter presents a review of the literature relevant to the development of the robotic weight-shift piloting system and the adaptive control techniques it employs. To provide a deeper contextual understanding of the thesis work, the review synthesizes and expands upon the literature references cited within each of the presented papers. The review is divided into several sections, each addressing a different facet of the research work.

Section 2.2 introduces model-based and adaptive model-based control approaches that are commonly used to manage the nonlinear dynamics prevalent in modern aircraft flight systems. A broad selection of control techniques addressing different aircraft control problems and other complex nonlinear dynamics' modelling issues are discussed. The integration of real-time adaptive mechanisms is often needed to compensate for the limitations of static models, highlighting their importance in the face of modelling uncertainty and dynamic environments. This sets the stage for understanding the challenges of implementing model-based control systems in aircraft, particularly when dealing with parameter uncertainty for nonlinear dynamics of WSC aircraft flight.

Section 2.3 discusses model-free control, a category of control techniques formulated without explicit knowledge of the system's dynamics or underlying models. These methods are useful when the dynamics are complex or poorly understood, as is often the case with nonlinear systems like the WSC aircraft. By using real-time data from system inputs and

outputs, model-free control methods adapt to changes and disturbances, providing a more robust alternative to model-based approaches affected by parameter uncertainties and sensitivity to linearization. This flexibility is crucial for managing the unpredictable dynamics of WSC aircraft, reducing dependency on accurate model parameters and enhancing system responsiveness to environmental disturbances.

Section 2.4 introduces the context for optimal control problems solved with approximate dynamic programming (ADP) techniques, which are well-suited for environments where traditional dynamic programming is impractical due to computational limits or model uncertainties. A subset of ADP methods employ reinforcement learning to identify optimal control policies directly from system feedback, bypassing the need for system models. By integrating online learning, ADP methods can adapt to changes in system dynamics and environmental conditions, and perpetually refine control strategies to ensure optimal performance. Applications of ADP techniques to various control problems are presented to showcase solutions that enhance system robustness and adaptability amidst competing control objectives, modelling uncertainty, and disturbances. This adaptability makes ADP techniques suitable for robotic control problems like the one posed by a piloting system for WSC aircraft.

Section 2.5 presents a review of existing literature on the dynamics modelling and control of WSC flexible wing aircraft. Given the niche interest for this aircraft class, the body of available research is less extensive than that for more conventional aircraft types. Nonetheless, there are substantial experimentally validated theoretical developments of dynamic models that capture the aerodynamic, aeroelastic, and WSC properties of hang-gliders, which are practically identical to air-trikes with respect to their dynamics. The section concludes with a discussion on recent advances using model-free control techniques that have been developed to address issues presented by modelling parameter uncertainty, which are particularly relevant due to the high degree of nonlinearity inherent to WSC aircraft

Section 2.6 details the literature relevant to cable-driven parallel robots (CDPRs), the primary actuation mechanism for the robotic weight-shift piloting system studied in this thesis. These mechanisms integrate the reduced weight and simplicity of cable-based actuators common to aircraft systems with the strength, rigidity, and high dynamic control performance typical of parallel manipulators. Applications of CDPRs are broad, from large-scale slow industrial manipulators to small, faster robots used for precision tasks. The kinematic and dynamic modelling for CDPRs is complicated by the nonlinear elastic behaviour of cables and the need for coordination among multiple actuators under dy-

namic loads, which presents challenges for achieving highly performant, accurate control and remains a primary focus of ongoing research.

Section 2.7 examines literature on open-source robotics simulation platforms with a focus on Gazebo, the chosen simulator for developing the dynamics model of a WSC aircraft in this thesis. The applications reviewed show Gazebo's ability to provide realistic environmental interactions which are crucial for validation of robotic systems prior to their deployment in the real-world. The active community continues to enhance its adaptability, as evidenced by the range of plugins developed for simulating and testing UAV controls that are discussed.

Section 2.8 provides a summary of the key literature that informed the thesis work.

2.2 Model-based and adaptive model-based control

Modern control theory is applied broadly to the multi-input multi-output control systems design for high performance aircraft [41]. Sophisticated controllers are required to effectively operate aircraft systems across a wide range of flight conditions while managing their inherent nonlinear dynamics. Aircraft control problems are often resolved by selecting suitable mathematically defined performance criteria. The optimal control gains are then computed to close all control loops simultaneously such that the performance requirement is met [42]. Standard approaches rely on idealized mathematical models to approximate the dynamics of the system. For instance, model-following methods aim to ensure that the aircraft behaves similarly to a predefined model by minimizing response differences. Similarly, dynamic inversion designs control systems for nonlinear behaviour using feedback linearization and a comprehensive model of the aircraft's nonlinearities. Robust design combines classical control theory with modern approaches, focusing on frequency-domain robustness analysis to assess system sensitivity across frequencies. Robust control techniques are commonly applied to nonlinear controls design for high performance military aircraft [43].

A common approach to address the challenge of nonlinear modelling is to discretize the dynamics into distinct operating regimes. Gain scheduling is ubiquitous for modern aircraft systems, e.g. the General Dynamics F16, or NASA X15 [44,45]. The method simplifies controller adjustments across different operating conditions by quantizing them into discrete detectable states. Sets of control gain parameters that are found for different operating

conditions are stored in a look-up table, significantly reducing the engineering effort that would otherwise be required to implement the full control system solution. However, this method requires sophisticated sensing of the flight conditions to detect timing for mode switches, which can induce operational errors such as the chattering effect that occurs when the control oscillates between condition regimes due to the discontinuous property of the discrete look-up table. Gain scheduling can be improved by the addition of robust techniques, as demonstrated for a vertical or short-take off attack aircraft reference model [46]. Sliding mode control employs a model-based approach to manage the nonlinear dynamics of aircraft by maintaining system states on a specifically defined set of conditions derived from the system's equations to ensure stability and robustness [47]. This control method effectively counters disturbances and variations in parameters with aggressive, discontinuous actions as shown with the F-16 longitudinal dynamics model [48]. Sliding mode control can account for systems faults, thereby improving robustness of the system, as shown for large Boeing 747 transportation aircraft [49, 50]. Alternatively, model-based fuzzy logic systems present another strategy for handling the nonlinear dynamics of aircraft and are well suited to manage modelling uncertainty [51].

Despite the sophistication of the control methods described earlier, they depend on idealized mathematical models which can cause sensitivity to model inaccuracies that adversely impact control performance when the actual aircraft dynamics deviate from theoretical predictions. Incorporating real-time adaptive mechanisms into modelling-based control design provides the essential capability of dynamic parameter adjustment. Flight control was one of the strong driving forces for the early development of adaptive control [52]. Åström and Kumar provide an excellent record of the historical development for adaptive control and optimal control in their survey of feedback control theory [53]. In the most general sense, an adaptive controller is simply a controller with adjustable parameters and an automatic mechanism for adjusting those parameters based on observations of its own performance. The study of adaptive model-based control strategies is wide-spread for nonlinear dynamics problem. Model predictive control (MPC) methods, while traditionally not adaptive, continue to be extended with adaptive features that enable controllers to refine a predictive model with real-time data, thereby mitigating the risk of model inaccuracies and enhance control efficacy [54]. An overview of modern MPC along with its applications to industrial processes, energy systems, power electronics, and manufacturing is presented in [55]. Adaptive MPC approaches are well-suited for integration with reinforcement learning to develop closed-loop control policies, yet they remain sensitive to modelling biases, as is the case for deforming robotic manipulators models [56]. Bruder et al. demonstrate adaptive MPC applied to soft robots with nonlinear behaviours and error-prone models that have physical simplifying assumptions [57]. This approach applies Koopman operator

theory to identify optimal linear approximations of nonlinear systems without the usual decline in performance associated with linearization, allowing for development of new control strategies in the presence of uncertain dynamics models. Model-based data integrated techniques have become more wide-spread for large-scale industrial processes [58]. While the large amounts of available process data provide insights into the operational dynamics, this approach requires automated tuning of sophisticated models and substantial computation effort in order to be useful. Chowdhary and Johnson demonstrate the capability of an adaptive concurrent model-based reinforcement learning scheme to tune a dynamic model inversion based control for an autonomous helicopter UAS [59]. Lyapunov stability analysis is used to show that adaptation scheme yields stable and bounded controllers. A similar solution approach is employed by Kamalapurkar et al. for the infinite-horizon optimal tracking problem presented in [60].

Model-based control systems have demonstrated significant capabilities for handling the complex dynamics of high-performance aircraft. The works highlighted present proven systemic design and analysis tools. However, their performance depends on accurate models. This is a major limitation especially when faced with uncertain conditions or system changes. Methods such as robust design, adaptive units, or fuzzy logic are essential to manage uncertainties for partially modelled systems [61]. Alternative approaches become practical when dealing with nonlinear systems characterized by uncertain conditions or frequent system changes, where the issues of inadequate models become more pronounced. This is particularly relevant for the control of WSC aircraft, where dynamic complexities and modelling challenges are significant. An option is presented by model-free approaches, which operate directly from a system's inputs and outputs, thereby sidestepping the issues associated with model-based control.

2.3 Model-free control

Model-free control is a class of controller design techniques that do not require explicit knowledge of the system dynamics or process model. Model-free approaches are best applied to control problems where model-based control strategies are impractical or ineffective due to dynamics that are unknown or difficult to accurately model. The complex nonlinear dynamics inherent to flexible wing weight-shift aircraft make model-free control techniques a compelling choice for the control of such systems [62]. As an example, Chaffre et al. employed a model-free control approach to design a micro UAV multi-copter controller that relied solely on sensor feedback and required no model of the aircraft dynamics [63]. The

solution enabled the micro UAV control to remain robust to wind gusts that the aircraft would otherwise be highly susceptible to. The model-free method adapted to the varying wind speed levels and optimized responses across different operating points, eliminating the need to be overly conservative to satisfy performance specifications. These results suggest model-free control would be suitable for a hypothetical flexible wing weight-shift aircraft flight controller that would need to account for velocity dependent dynamics and susceptibility to wind disturbances without being too conservative on control. There are several proven approaches that fall under the category of model-free methods, including model-free predictive control, the widely studied methods of fuzzy logic, and reinforcement learning.

Model-free predictive control methods use measured input and output data to predict a model for the systems operation over time, thus remains sensitive to changing conditions while ensuring optimality of the input action [64]. These methods are shown to approach the performance of model-based methods for the different control applications of power converters, electric drives, and microgrid systems. Wojsznis and Blevins discuss several self tuning intelligent PID control schemes in [65]. The model-free methods discussed detect a change in the monitored process variables time dynamics, and adapt either by rebalancing the proportional and integral correction signals, or by switching between different gain combinations from a predefined set according to a performance index. Fliess and Join present a rigorous treatment of design for intelligent PID controllers in a model-free way using ultra-local output models [66]. Similar in principle to a self tuning regulator, this method reduces the PID gain tuning problem to a regression on the output tracking performance of the system. The ultra-local approach has been shown to achieve superior reference tracking and disturbance rejection even for fast power electronics systems [67]. Madadi et al. compare intelligent PID controller performance with two model-free predictive control techniques [68]. It is found that the intelligent PID approach offers better disturbance rejection and tracking performance.

Fuzzy logic control systems were first invented as a model-free approach that relied on expert knowledge of a systems dynamics. These methods classify the system's dynamical state and corresponding control into membership categories that are human-interpretable. A recent review of their historical development and future directions can be found in [69]. For systems with greater degrees of uncertainty, Type-2 fuzzy logic control systems are of interest as they allow the membership functions to incorporate confidence levels [70]. The primary issue with model-free fuzzy logic controllers comes down to the fact that it is essentially heuristic, which makes their design and improvement processes strongly designer and system dependent. A well-used tool to tune fuzzy logic controllers is through neural

network training, where the adaption of membership sets is driven through performance evaluation based on episodic data [71, 72]. Neuro-fuzzy logic controllers promise faster training and improved performance, but require substantially more technical overhead and a method to generate training data.

Nonlinear function approximation using neural networks is an important addition to model-free control. A learning neural network compensator unit can be added to a control system to improve performance under time varying and unknown nonlinearities that are otherwise difficult for the model-free methods to account for. Examples of application of this method are shown for a continuous stirred tank reactor in [73], and for an aircraft pitch control mechanism in [41]. Das et al. demonstrate the application of a learning compensator unit in [74]. El-Sheimy et al. use a neural network approach to fuse inertial navigation data and GPS data without any requirement for a dynamic model of the vehicle that is normally used for position estimation problems [75]. It is found that the method outperforms a Kalman filtering routine that was used to benchmark the results.

On another front of model-free control via neural network function approximation is the application of reinforcement learning. Rather than compensating an existing controller, entire nonlinear feedback controller structures can be learned with neural networks from input and output data alone by allowing the controller to interact with the system and adapt through reinforcement learning. Barto, Sutton and Anderson were among the first to present an early example of this type of model-free control with the cart-pole dynamics model [76]. The reinforcement feedback signal in this work solely relies on the binary occurrence of failure events and otherwise contains no information of the system state. Consequently, the adaptation of the neural network is predicated on the aggregate performance of the system rather than modifications to individual elements of the control process. Cook demonstrates that a simple two-neuron network can control the non-minimum phase dynamics of a bicycle system in a desired direction without algebraic analysis or extensive learning [77]. Deep reinforcement learning extends the capabilities of reinforcement-based learning controllers by leveraging high-dimensional information from the system state [78]. Performant model-free controllers can be learned simply by employing rich observation data, such as camera feeds focused of robotic action or pixel data from video games,

Reinforcement learning based neural network approaches have been applied to solve optimal control problems with complex nonlinear dynamics interactions. For instance, they have been extensively applied to the coordinated actions of multi-agent systems, where interactions between agents must be accounted for [79]. These solutions embed optimality conditions, rooted in game-theoretic principles, in the reinforcement process to enhance

collective decision-making and performance of the resulting model-free controllers used by the multi-agent system. A similar game-theoretic framework is used for the training of neural network based solutions for robust H_∞ controllers [80]. Significant to this thesis is the branch of optimal control that involves ADP-based techniques that incorporate online reinforcement learning and optimality conditions to extend the capabilities of model-free approaches to complex nonlinear systems with partially unknown or fully unknown system dynamics.

2.4 Optimal control by approximate dynamic programming techniques

Optimal control frequently involves solving complex matrix design equations with adaptive techniques to obtain real-time approximate solutions where exact methods are impractical [62]. The results from the prior work of Abouheaf et al. demonstrate several partially model-based and model-free control techniques applied to flexible wing weight-shift aircraft control [22–27, 81]. Among these, methods that employed approximate dynamic programming (ADP) schemes were found to be noteworthy. These methods use reinforcement learning to find optimal control strategies online from feedback data, using either partial model-based techniques or completely model-free.

Dynamic programming, first introduced by Bellman in the 1950s, is an optimization approach that simplifies complex problems by breaking them down recursively into smaller, manageable sub-problems [82]. These sub-problems are then solved sequentially, and by Bellman's principle of optimality can be applied in a forward or backward sequence depending on the problem structure. For optimal control problems, these methods use a necessary optimality condition known as the Bellman equation that defines the value of a current control input as the expected reward along with the predicted value of future sub-problems. The optimal control strategy is then found by iteratively solving for control solutions that maximize total value accumulated from all sub-problems. Solving this Bellman equation for the case of linear dynamics and quadratic cost leads to a Riccati equation whose solution directly gives the optimal feedback gains. Dynamic programming has two important limitations worth mentioning: First, since the solution is found offline by solving all sub-problems at once, it is model dependent. Second, the approach becomes infeasible for high-dimensional problems or continuous time problems that have the exponential growth in the state and action spaces, where the amount of data required to obtain

an accurate model also increases exponentially. This phenomenon, known as the *curse of dimensionality*, is a simple fact for most real-world time-dependent robot control problems.

ADP methods were developed to address the limitations of classical dynamic programming. ADP is a body of techniques that combines reinforcement learning with dynamic programming to determine optimal control policies in problems characterized by model uncertainty [83, 84]. Werbos first advanced approximate solutions to dynamic programming problems that used automatic neural network training by backpropagation [85–88]. The online approaches generate approximations of value functions that encode state-action pair utility, (or *quality*), effectively prescribing the optimal control policy by observing the temporal difference (TD) error of probabilistically-derived models [89]. This model-free process came to be known as Q-learning [90]. The combination of automatic backpropagation and Q-learning, often referred to as *neuro-dynamic programming*, lead to a multitude of solutions to classic dynamic programming that were otherwise intractable due to *the curse* [91, 92]. Doya extended TD learning methods to continuous time and state problems in [93]. Optimal control problems once thought to be intractable were now addressable by learning from feedback data alone [42, 94]. Lewis et al. demonstrated the applicability of these techniques to robotic systems with unmodelled dynamics by learning trajectory tracking control policies purely online and in real-time [95, 96]. Khan et al. classify the existing types of TD methods into three types: Q-learning, state-actor-reward-state-actor (SARSA), and actor-critic [97]. Q-learning is off-policy, meaning that it estimates the value function independently of the current policy, i.e. adaptation does not occur during exploration. In contrast, SARSA is on-policy meaning that it adapts the policy during exploration and estimates the value function according to the state-action pair and the current policy. Actor-critic methods are on-policy, with the significant distinction that the policy and value function approximations operate interdependently, enabling simultaneous estimates to both components in a dual learning approach. Landelius presents proofs for the convergence to optimal control policies by different adaptive critic methods in [98]. Prokhorov provides nuanced discussion and contrast for increasingly capable adaptive critic designs in [99].

Of specific relevance to the thesis is the branch of ADP methods that employ online reinforcement learning adaptive critics schemes to solve the optimal control policy for discrete or continuous times problems with partially unknown or completely unknown dynamics [62, 100–102]. In this framework, control problems are solved by optimizing the performance of the underlying dynamical systems, encoded by so called Hamilton-Jacobi-Bellman (HJB) equations which define objective (or cost) functions that evaluate the system's evolution over time, based on feedback data. The HJB equations provide a

structure for optimal control problems where solving the costate equations derived from the Hamiltonian formulation is equivalent to solving the value function of a dynamic programming problem. These are solved iteratively with a reinforcement learning approach composed of two steps: The evaluation of the TD equation to determine the value function costate, followed by calculation for the optimal strategies to maximize the predicted value accumulated in time. The solution of the resulting TD equations depends only on the state or state-action combinations and allow for gradient-descent solution approaches. Lewis, Vrabie, and Vamvoudakis review the interrelation of several discrete time and continuous time approaches to optimal adaptive control via reinforcement learning in [103]. Balakrishnan et al. review the status of work towards closed loop stability guarantees for adaptive critic methods in [104]. In the case of online adaptive actor-critic approaches, two neural network structures learn by their interaction. The actor neural network approximates the optimal control policy, and the critic neural network approximates the value function that is used to evaluate the effectiveness of the actor. Al-Tamimi et al. give convergence proofs for online actor-critic policy iteration and value iteration in the discrete time case in [105], and further demonstrate that these methods can be made robust and reject disturbances in [106, 107]. Vrabie and Lewis give convergence proofs for online actor-critic policy iteration algorithm with unknown system drift dynamics in the continuous time case in [108]. Lewis and Vamvoudakis present a policy iteration and value iteration derived forms of a model-free output feedback approach for a partially observable system in [109]. Stability and convergence proofs for reinforcement learning based adaptive optimal controls with input saturation constraints for partially unknown systems are presented in [110]. This work is further extended to solve the linear quadratic tracking (LQT) control problem with entirely unknown system drift and reference trajectory dynamics [111]. Q-learning methods for tracking control for systems with completely unknown input and system dynamics are presented in [112–114], along with proofs of local stability for the error tracking and convergence to optimal policy presented in [115]. Adaptive critic methods have recently been extended to safety critical reinforcement learning problems. Yang et al. present actor-critic methods that provide safety guarantees on the reinforcement learning process by incorporating full state and input barrier function constraints [116]. Yazdani et al. developed a policy iteration scheme that guarantees policy updates respect system state and input bounds [117].

There are numerous demonstrated applications of ADP for control problems found in the literature. Naeem et al. provide a comprehensive overview of different applications for reinforcement learning in [118]. Si and Wang present an action dependent heuristic dynamic programming based solution to control a highly unstable cart-pole with three links, that is trained online without a model through TD learning [119]. Khan et al. apply a Q-learning

approach to design a model-free control for a humanoid robot arm [97]. Vrabie et al. apply an online actor-critic policy iteration to develop a power system load frequency controller [120]. Vamvoudakis and Lewis apply online actor-critic policy iteration to identify a feedback elevator control for linearized F-16 longitudinal dynamics [121, 122]. Lyu and Cheah leverage a combined offline-online data-driven learning approach to alleviate concerns around kinematic and dynamic uncertainty for a robotic manipulator by embedding into the control an offline trained neural network that maps task space end effector commands to joint space commands, and then modify these weights during operation through online reinforcement learning [123]. Arogeti and Lewis present several output feedback based lane keeping lateral steering controllers that are found with offline and online methods and used only partial system information [124]. Mazouchi et al. present a conflict aware output feedback reinforcement learning controller for an autonomous vehicle that dynamically modifies the reward function in real-time according to a higher level attention mechanism system [125]. ADP-based optimal control methods scale well to multi-agent system problems such as mobile robot coordination control. Qu et al. developed a flock motion control solution that resolves competing objectives of leader trajectory following, swarm consensus motion, and neighbour separation motion [126]. The leader tracking control of each mobile agent is determined by online policy iteration. Adepegba, Miah, and Spinello developed optimal swarm coordination feedback laws for the area coverage control problem with actor-critic policy iteration [127]. Soleymani et al. extended this approach by considering value function forms that directly incorporate the area coverage metric [128]. Inter-agent interactions may be defined within the framework of graphical games where the communications between the nodes are described using graph structures. [129–139]. The optimal solutions generated rely upon the interdependence of the value functions assigned to each agent and their interactions with either neighbouring units or the total swarm.

The body of work reviewed underscores the versatility of ADP methods in addressing adaptive control across diverse applications. ADP and online reinforcement learning solutions by their innate structure, require continuous activity and exploration to refine control policies in response to the dynamics of evolving systems that may possess unmodelled uncertainties. As Barto, Sutton and Anderson point out, the evaluation driven nature of reinforcement learning makes it a more general optimization process than conventional error correction based optimization techniques that only seek equilibrium in the system [140]. The continuous need to explore and adapt towards further improvement enables ADP methods to generate sophisticated, real-time solutions for complex and high-dimensional problems, thus effectively address real-life optimal control challenges. Given this inherent property of ADP-based optimal control methods to continuously refine control policies in the face of unmodelled uncertainties and evolving system dynamics, they are particularly suited

for addressing the nonlinear dynamics of WSC aircraft, where traditional modelling faces significant challenges.

2.5 Approaches for modelling dynamics and control of weight-shift controlled flexible wing aircraft

The academic literature on WSC flexible wing aircraft is limited compared to more conventional aircraft types. The constrained market demand resulting from the limited applications of flexible wing aircraft restrict their commercial viability, leading to fewer funding and development opportunities [6,8]. Accurate dynamic models for these aircraft are challenging to develop as they possess nonlinear dynamics due to the flexible wing's aeroelastic deformation, as well as two-body weight-shift relationship that is inherent to their flight control.

Flexible wing WSC aircraft trace their roots to engineering designs developed by Rogallo, with the original intended application of recovering space vehicles after atmospheric entry [141]. As they became adopted for flying, the following design evolution of hang-gliders was done on a trial-and-error basis, absent of scientific rigour, resulting in poorly understood stability properties and high accident rates [9,18,142]. Rogallo initially worked on flexible wings privately, later contributing to the founding of NASA's Flexible Wing Section at Langley Research Center in 1958 [141]. Their design can be thought of as existing on a continuum between parachutes and conventional rigid wings. The aerodynamic shape of the wing is achieved by billowing of the fabric under dynamic pressure, exactly like that of wings in nature. These wings are notable for their simple construction that depends on careful additions of minimal structure to stiffen the fabric surface, thus constraining local deformation. The combination of ultra-low fabrication cost, lift to drag ratios approaching 15:1, and ease of maintenance provided by these wings led to the founding of the hang-gliding sport. Wing designs evolved with the sport, however the progress made had been based on practical trial and error experience, with minimal theoretical backing.

In 1979, NASA's Ames Research Center began a program with the aim to develop quantitative tools to determine the aerodynamic characteristics of flexible wing ultralights. At the time there only existed wind-tunnel studies for the original Rogallo wing design, and none for the modern hang-glider wings of the time [143]. The flexible wing aircraft industry was mistakenly relying on standard Rogallo wing data with high billow and low aspect ratio

of 2.5, while modern hang-gliders had far larger spans with aspect ratios of 5-8, reduced billow, and swept wings. The difference in dynamic performance and control relative to Rogallo wings became pronounced. 30% of fatal accidents were attributed to tumbling dives induced by wing luffing in low angle of attack at higher speeds, where conventional pilot training had mistakenly suggested safe operation. This mistake in engineering assumptions was killing people. The work reported by Kroo in [144] used a 1:5 scale wind tunnel experiment to analyze the relationship between aeroelastic deformations and the resulting longitudinal stability and control by considering variations in the wing's sweep, twist, dihedral and reflex by variations of the luff-line lengths, batten flex, reflex, keel pockets, and dihedral. Kroo points out that it is difficult to scale the aeroelastic properties of the flexible wing, as such this contributes to modelling error of the approach. Notably, the test found large destabilizing moments at negative angles of attack at higher speeds associated with poorly placed pilot weight-shift CoG. The hang-gliding sport community mistakenly assumed simple static conditions on the weight-shift mass from lift point were sufficient to ensure stability, and neglected the influence of airspeed, ergo, the effect of dynamic loading on stability and control. Kroo uses the experimental data to define small perturbation models for the longitudinal and lateral dynamics, and from this determines stability and control bounds [18]. He concludes this work with the statement "the simplicity of hang-glider designs belies the complexity of the analysis required to predict their behaviour". Put simply, the analytical methods used for conventional aircraft aerodynamics fails to predict basic dynamic characteristics of flexible wing weight-shift aircraft.

Coincidentally near the starting time of the Ames Research Center program, Cranfield Institute's College of Aeronautics, under Cook, began to study the aerodynamics and stability properties of hang-glider wings. These early efforts are summarized in [145]. Their experimental approach used ground-vehicle attached flying rigs to produce steady conditions. At first, the experiments were performed to determine the equations of motion for steady state flight, and to examine longitudinal static stability properties. Kilkenny continued this work by validating the mobile test rig data acquisition equipment with wind tunnel for a rigid wing section [146], and then with a focus on the effects of pilot control on the longitudinal stability [142, 147]. Information about the drag of a hang-glider pilot from wind tunnel test was used in conjunction with aerodynamic data gathered from the mobile flying rig for three hang-glider wings. Important facts regarding the sensitivity of the longitudinal stability were identified. Shifting the pilot's hang point attachment by small 10 cm increments was found to have a great destabilizing effect. Further, it was found that the stability of hang-gliders designed for pilots weighing 70 kg to 90 kg significantly decreases with lower pilot weights, leading to dangerous flight behaviours such as unstable transitions and inability to trim, especially at lower incidence angles and

lighter weights below 40 kg. The effects of unsteady flight conditions on WSC were also considered. While upward gusts increase the effect of control, downward gusts dangerously reduce the effect of weight-shift control or can cause control reversal when strong enough. More recently, Gratton and Newman modelled the tumble departure mode for WSC aircraft that often leads to fatal accidents, with the resulting agreeing with the earlier findings by Kroo and Kilkenney [148]. These facts underscore the presence of nonlinearity in the flight dynamics and control of hang-gliders that had been underestimated by the sport's community. Greater emphasis on modelling the interplay between weight, balance, and aerodynamic stability is necessary to ensure safe operation.

The next major development in modelling the flight dynamics of hang-gliders began in 1989 when de Matteis developed a dynamical system model that considers a rigid wing along with a pilot that is attached by a fixed member to a rotation point on the keel [29, 149]. An important working assumption is made that aeroelastic effects can be added as velocity-dependent aerodynamic derivatives. The model is developed for small-perturbations around the trimmed states, and from this transfer functions are developed for the longitudinal and lateral response of the hang-glider to pilot input. The model correctly replicates the high sensitivity of longitudinal stability due to CoG shifting induced by the pilot. It is found that the system's response to a lateral shift of the centre of mass is relatively small. This implies that while the lateral stability margin is large similar to what is found in [18], maintaining the system's lateral stability is strongly dependent on sustained, disciplined pilot input, and control authority is sensitive to disturbances from gust. The frequency response of the hang-glider to pilot force is further considered in [150]. The model captures stabilizing damping effects attributed to unsteady circulatory airflow that is observed in larger angle of incidences, confirming that the hang-glider possesses positive static stability in the longitudinal axis. The effects of turbulent air on the hang-glider's longitudinal dynamics for this model are considered in [151]. A similar effect was experimentally confirmed more recently with scaled wind-tunnel test performed by Khaddage [152].

In the 1990's, Cook's research group had continued on with the work to develop dynamics models that considers the aeroelastic effects of the flexible wing along with the pilot control as detailed in [19]. First, a mathematical model for the longitudinal static stability of the hang-glider was developed based on the previous work by Kilkenny [153]. This model was validated by fitting the model output to reference data gathered from a fifth generation hang-glider wing, and it is found to be in broad agreement with the results of Kroo and de Matteis, despite being arrived at separately. Next, the model was extended to consider the aeroelastic properties of the flexible wing. The dynamic stability properties of the hang-glider were addressed at first by Rollins who considered the effect of velocity dependence

on longitudinal stability and control [154]. A semi-empirical approach is used to identify mathematical expression for the aeroelastic properties of the wing that depend on the variable camber and twist when near trim in steady flight. This approach involves dividing the wing into multiple chord wise segments. The aerodynamic properties of the wing are determined by how the span-wise twist and section lift coefficients change with changes in incidence, velocity, camber, and the action of the luff lines. Spottiswoode then developed comprehensive computational tools to model the longitudinal and lateral dynamic stability and control of the hang-glider, factoring in the aeroelastic properties of the flexible wing [155]. The work includes consideration for the dynamic stability properties of the hang-glider when factoring in the effect of velocity dependence on stability and control. This leads to what can be considered as the state of the art for sophisticated modelling of the flight dynamics of the hang-glider, reported in [19].

The most advanced mathematical model developed by Cook and Spottiswoode in [19] consist of a system of nonlinear dynamic equations that describe all kinetic phenomena of the hang-glider. The modelled dynamic properties include the force and moment balances for inertial, gravitational, and aerodynamic lift and drag, as well as the trim equilibrium conditions. Small aerodynamic and pilot-induced disturbances in the motion variables are represented by a set of stability and control derivatives by techniques that match those found in classical flight control textbook treatments [156]. The most pertinent aspect presented is the complicated treatment of the velocity dependent aerodynamic model for the deformable flexible wing. This approximate model replicates the main features of the deforming airfoils aerodynamic behaviours while ensuring that it is clear which physical phenomena are driving said behaviours. The primary deformation features modelled map the changes in the wing's luff, camber, and twist to corresponding effects on lift and drag, and then uses these to calculate the aerodynamic trim, stability and control derivatives. The overall stability properties for the aircraft and response to control inputs of the pilot for longitudinal and lateral axes are then found. In the authors' own words, the approach is chosen as it represents a compromise between simplicity and accuracy. It is limited in that it is quasi-static about equilibrium trim conditions and does not account for unsteady aerodynamic or shifting mass effects, both facts of life for real-world WSC flight. That being said, this model provides the most comprehensive representation, to date, of the complex nonlinear dynamics governing WSC flight for flexible wing ultralights. One would have to account for this level of dynamic complexity when designing a hypothetical automatic feedback controller by model-based approaches. Several important conclusions on the flight dynamics are drawn from this work that influence future automatic controls design:

- Aerodynamic and control derivative estimates are highly dependent on the modelling data used.
- The WSC pilot creates a pendulum stability property which increases the degree of static stability in both pitch and roll.
- The WSC pilot pendulum greatly amplifies the yaw stability provided by the swept wing.
- The phugoid mode becomes unstable at low flight speeds due to the wing's aeroelastic properties, which reduce effective damping.
- Wing flexibility causes significant variations in the pitching moment with speed, leading to a destabilizing effect at lower speeds and improving stability at higher speeds.
- At higher speeds, the aeroelastic effects become less variable, i.e. the wing is more rigid.
- Longitudinal dynamics are highly sensitive to WSC.
- Lateral dynamics are comparatively lowly sensitive to WSC.
- Lateral control must account for the instantaneous adverse yaw input that is an unavoidable consequence of a roll input, but this resolves naturally after a short time.

In 2015, Ochi developed further on the Cook and Spottiswoode model by considering the system as two coupled bodies, with internal forces to represent the weight-shift interaction. At first, Ochi only considered the longitudinal dynamics [157]. Simulations revealed the two-body model was unstable in most flight conditions, prompting substantial adjustments to pilot and wing aerodynamic derivatives to arrive at stabilized flight performance that aligned with the original one-body model. Ochi considers the lateral flight maneuvers for the two-body model in [30]. Numerical simulations confirm that during turning, the pilot stays near the wing's symmetry plane with minimal control effort, mirroring the actual flight principles of WSC. Ochi's modelling work culminates with a two-body system nonlinear dynamics model that fully describes the flight dynamics as well as the pilot's WSC input [20]. The pilot's handling is modelled as linear feedback controller that performs three actions to stabilize the aircraft's dynamics: First, a pair of pitch and roll attitude control loops determine the wing body maneuver, with yaw neglected. Second, a stability augmentation system controls the rotational (yaw) state of the pilot body relative to the wing to ensure stable and coordinated motion, preventing unintended rotations and enhancing overall flight safety. Third, a control augmentation system responds to pilot inputs to facilitate lateral shifts of the CoG, regulating it to be vertically aligned with the wing body's CoG. Numerical simulation results for the model confirm that much like real-life,

the hang-glider without control is unstable except for the case of perfectly trimmed, steady flight. When the feedback controller is implemented, the aircraft model is shown to become stabilizable for varying flight conditions. With air-trikes, the wing yaw control is addressed by successive roll attitude adjustments through roll-yaw coupling, as the pilot mass is not able to rotate freely in yaw as is the case for a hang-glider [8]. This indicates that any effective weight-shift flight control system must manage these three critical functions of the pilot-wing system.

Significant challenges for model-based control include aeroelastic deformation of the flexible wing, turbulence-sensitive stability, and the nonlinear response of weight-shift maneuvers, which depend on the interplay between the vehicle configuration and the flight envelope phase. It is difficult to measure all of these effects in experiments, ergo, higher uncertainty for modelling-based control approaches is expected as concluded by the model developers [18–20, 142, 149]. Abouheaf et al. applied model-free reinforcement learning approaches to control the Cook-Spottiswoode and Ochi hang-glider dynamics models. Model-free control provides a method to overcome issues caused by nonlinearity inherent to the hang-glider system's dynamics and the higher degree of parameter uncertainty of models for such a system.

Abouheaf and Gueaieb explored several model-free control methodologies. Initially, they considered the application of an adaptive actor-critic learning structure to control the flight dynamics of the linearized Cook-Spottiswoode model [23, 24]. The method described in the work is partially model-free as it assumes no knowledge of the drift dynamics but requires a model for the system's inputs. A policy iteration scheme updates an approximation for the costate which encodes the control system's optimal performance for stabilizing the system. The online reinforcement learning algorithm employed generates estimates for the system's optimal value function by sampling measurements in an online fashion, and periodically updates the control policy. Simulation results present a learned candidate optimal feedback controller that stabilizes the system's longitudinal and lateral dynamics. Further, this method is shown to be robust to substantial random noise added to both the simulated drift and input dynamics, as well as to noise added to the measurements of the longitudinal and lateral state feedback.

Abouheaf and Gueaieb considered the application of adaptive fuzzy logic structures to the tracking problem for the lateral dynamics of the linear Cook-Spottiswoode model in [25]. Although model-free, the method requires the heuristic definition of fuzzy logic rules that implicitly map state error measurements to control inputs. The fuzzy logic weights are updated using two different reinforcement learning methods, a Q-learning approach and a

neuro-fuzzy approach. Simulation results demonstrate that both learning methods are able to provide a optimal trajectory tracking and maintain the stability of the lateral dynamics while exceeding the performance of a static fuzzy logic controller. These results do not consider the longitudinal dynamics, so it is left open as to how the approach would perform for simultaneous longitudinal and lateral trajectory tracking.

Abouheaf et al. proceeded to develop fully model-free online reinforcement learning controllers to solve the optimal control problem for the Cook-Spottiswoode and Ochi hang-glider dynamics models. No model for the input or drift dynamics is required with these methods, and they are inherently robust to parameter uncertainty and disturbances. Each solution uses online adaptive critic structures to approximate optimal value functions and the corresponding optimal control policy by sampling the system's performance over time. First, an adaptive critic integral reinforcement learning approach that adapts via value iteration is applied to the Cook-Spottiswoode model [81]. The critic adapts according to a gradient descent rule, while the actor control policy is defined directly by the critic. Stability analysis shows that the method is able to regulate the lateral and longitudinal dynamics. This approach is shown to control and stabilize the dynamics of the more complicated two-body Ochi model [22]. Methods with separate actor-critic neural network structures were then studied. This is first applied to control the lateral dynamics of the Cook-Spottiswoode model, and solved for dual-objectives of tracking error correction and dynamic stabilization using value iteration [26]. This approach was employed to control both the lateral and longitudinal dynamics in [27].

While these recent developments using model-free control methodologies have demonstrated potential for managing the uncertain nonlinear dynamics of flexible wing weight-shift aircraft, significant challenges persist in translating these approaches to hardware implementations. The inherent complexity of real aircraft behaviour necessitate further development to ensure the applicability of these control techniques in real-world experiments. Additionally, the lack of a defined actuation mechanism to implement these controllers introduces further complications, as the control and modelling of a hypothetical robotic weight-shift piloting system would need to be incorporated.

2.6 An overview of cable-driven parallel robots design, dynamics, and control

A cable-driven parallel robot (CDPR) is the primary actuation mechanism employed for the robotic weight-shift piloting system detailed within the thesis. They are composed of a manipulator scheme that uses a set of parallel flexible cables or wires under variable tension from actuators that drive the movement of an end-effector. These robots offer high end-effector precision, variable compliance, higher overall structural stiffness, and are well-suited for applications with larger workspaces [158]. They combine desirable flexibility, lower mass, and simplistic design properties of cable-driven manipulators with the high precision and strength of parallel manipulators [159–162]. Zarebidoki et al. published a comprehensive review regarding kinematics and dynamics modelling, configuration optimization approaches, and control for CDPRs [163]. The review identifies a significant gap in the current research on CDPRs with respect to modelling of nonlinear elastic effects of the cables while under static or dynamic loading, along with the high potential for state uncertainty. It is suggested that further study should focus on enhanced CDPRs that employ new hypothetical adaptive control systems that more easily manage actuator reconfiguration, as this would complement the modularity and scalability of CDPRs. Pott et al. provide a brief review of existing CDPR prototype development in the literature along with a full systems design overview for the Fraunhofer IPAnema CDPR [164]. This work includes details on the mechanical design for a servo-controlled winch with cable-guide mechanism, their approach to dynamic simulation, and experimental results for the compliance and stiffness of the CDPR when the end-effector under load. Francois et al. present a comprehensive overview for the typical design process in [165]. This work presents a complete treatment of the systematic design approach taken to select a correct kinematic structure, sizing and placement of the parallel actuators, the optimization of design parameters, and the dynamics modelling required to ensure suitability for the industrial application for a single 4-DoF manipulator. Starting out as an academic effort, the authors found that their design is competitive with the state-of-the-art robots commonly used in industry through simulations, leading to prototype manufacturing and experimental test. This work underscores the fact that there remains rich opportunities for innovation for these robots even with simple improvements on designs based on first principles.

The kinematic capabilities of CDPRs depend on joint configuration and the inclusion of constraining mechanisms. Cable-driven parallel manipulators are divided into 3 types depending on the n DoFs, driven by m cables: under-constrained when $n + 1 > m$, fully constrained when $n + 1 = m$, and redundantly constrained when $n + 1 < m$ [166]. The

kinematics of under-constrained CDPRs are significantly influenced by external forces acting on the end-effector, as they have fewer cables than degrees of freedom. Carricato and Merlet provide an algorithm to assess the stability of equilibrium states for crane-type CDPRs [167]. Extensions of parallel manipulator dynamics models to cable-driven parallel manipulators is an active field of research. Dynamic analysis for CDPRs is better understood by first studying the dynamics of parallel manipulators with rigid links. As an example, the parallel manipulator dynamics analysis in [168] presents a Lagrange-d'Alembert formulation as well as an Newton-Euler formulation, and proves their equivalence. The statics and inverse kinematics of a fully constrained cable-driven parallel manipulator, along with its inverse Jacobian relationship is presented in [169]. A full dynamics model derived by Newton-Euler formulation for an under-constrained cable suspended robot is presented and experimentally verified in [170]. A Lagrangian energy formulation is used to derive the dynamics model for a fully constrained 6-DoF CDPR along with the Jacobian relationship is calculated in [171].

CDPRs possess several notable technical issues in their real-life application. Practical issues include performance sensitivity to cable elasticity, difficulty coordinating parallel manipulators, and end-effector sensitivity to modelling uncertainty and disturbances. Despite these issues, the use of cable-driven parallel manipulators has increased significantly in recent years. In their review, Qian et al. found many studies that have focused on investigating different aspects of CDPR designs, including their typical configurations, dynamic analysis techniques, control methods, and applications [172]. A significant degree of research effort is dedicated to the optimization behind the mechatronic designs for parallel manipulators. The challenge of optimizing actuator placement or end-effector cable anchor points for accessible workspace and manipulability properties is a pervasive problem, as there is an inherent trade-off between the two objectives. Ebert and Voglewede discuss conditions on force closure problem for cable-driven parallel manipulators, and identify mathematical connections to multi-finger grasping [173]. The authors also consider tension force feasibility for wrench-feasible workspaces, static equilibrium under gravity, dynamic equilibrium, and statically stable poses. Miyasaka et al. provide experimental validated hysteresis model for stainless steel cables with different diameters in [174]. Paty et al. perform a end-effector pose error sensitivity analysis for elastic and geometric parameters of a CDPR which incorporates a novel double revolute (2R) joint pulley design in [175]. The sensitivity analysis is used to determine the optimal configurations that lessen trajectory tracking error among several CDPR designs. Of note, it is found that the extra degree of freedom provided by the 2R pulley design leads to improved overall trajectory tracking performance by increasing the tracking-error-limited volumes of the workspace. A similar outcome was observed for a rigid parallel robot in [176], where the workspace

was enhanced by introducing additional free motion through a nonholonomic joint added to the actuator mounting point. Khakpour et al. study the idea of replacing single cables with spatially distributed sets of cables that are all driven by the same actuator, i.e. a differential mechanism, to improve redundancy under higher end-effector loading [177]. It is found that for identical per-actuator force distributions the differential CDPR scheme offers improved force-closure and wrench-feasible workspaces than the conventional single cable per-actuator scheme. However, this solution requires substantial additional mechanical and control sophistication to ensure that differentials are loaded equally. Barbazza et al. present a CDPR with an online reconfigurable end-effector that allows for the workspace and manipulability to be transformed through modification of the cable anchor points [178]. This scheme allows the control designer to more readily consider trade-offs between pose transitions and achievable wrench forces of the end-effector.

Motion controller designs for parallel manipulators are an active research topic due to the increased difficulty of modelling their inverse kinematics and deriving control laws, relative to serial manipulators. For CDPRs, commanding the end effector to track a given trajectory requires the simultaneous coordinated control of multiple actuator's torque-tensioning commands [179, 180]. Current research is primarily focused on offline dynamic trajectory planning, where an inverse kinematic model is required to solve coordination problem of the actuators [181–185]. As these are model-based approaches, they suffer from the issues posed by model uncertainty or disturbances that may act on the system. There exist a variety of research efforts that instead employ feedback control and data-driven approaches to design trajectory tracking controls for parallel robots. Fahham and Farid present a clever feedback linearization approach that exploits optimal control theory to devise a bang-bang controller and solve for the corresponding switching points in [181]. Ghasemi et al. implement a neural-network training scheme that learns the forward kinematics model directly from measurement data in [186]. Korayem et al. employ a robust feedback linearization approach that finds optimized trajectories through a linear quadratic regulator scheme in [187]. Model-based approaches based on the Udwadia-Kalaba constrained dynamics formulation are applied to the trajectory tracking control for several different parallel manipulators in [188–192]. Sancak et al. apply a model-free approach to the tracking control design of a CDPR with partially unknown dynamics, where a nominal dynamic model in feedback linearized form is equipped with an ultra-local model predictive feature to self-tune an adaptive PID controller [193]. Jabbarias and Yoon present an adaptive and robust model-based approach with guaranteed trajectory tracking performance, where a leakage-like adaption law is employed to modify the control input in the presence of significant disturbance or modelling uncertainty [194]. This work is extended to employ a neural-network training scheme in order to learn a model-free trajectory tracking controller

in [195]. Mann and Surgenor apply a model-free fuzzy logic based PID controller to a rigid parallel manipulator in [196]. The offline deep learning approach presented [197] employs neural networks to learn unknown dynamics properties of CDPRs. The discussed works offer valuable insights into the progress and challenges involved when developing CDPR hardware and controls, and will inform development for the weight-shift flight control piloting system.

2.7 Open-source mobile robotics simulation, with a focus on Gazebo

To achieve the development of high-performance mobile robots capable of fulfilling sophisticated missions, computational platforms that enable dynamical simulation have become essential. These simulation tools are critical for evaluating robotics designs, as the combination of virtual dynamics model and ability to move in a simulated environment enables rapid validation of prototype iterations without incurring real experiment cost. [198, 199]. There is a growing focus on investigation of software platforms, and simulation tools for mobile robotics specifically, as renewed commercial interest has increased their demand. It is important to recognize that simulation software evolves more rapidly than other facets of the robotic prototyping process. Therefore, the discussions of robotics simulation software presented here should be viewed with the understanding that the literature and tools discussed represent only snapshots at the time of writing, and they cannot capture swift developments in the field. In 2009, Parodi et al. presented a survey of hardware in the loop simulators with special attention given to their applicability to multi-vehicle marine scenarios [200]. In 2014, Ivaldi et al. presented a survey for different humanoid robot dynamics simulators that incorporates significant user feedback within the analysis [201]. The survey found that the majority of robotics physics simulators for the time did not address key roboticists' defined requirements of support for mathematical descriptions of multi-body systems and optimized contact solvers. Gazebo (presently referred to as Gazebo classic) and V-REP (since renamed as CopelliaSim) were the preferred options that met key criteria of roboticists. Recently, the exponential growth in research and commercial activities involving machine learning for robotics control tasks has precipitated the need for simulators that can support these developments. In 2020, Zhao et al. presented a literature survey on sim-to-real deep reinforcement learning methodology and software for robotics applications [202]. These approaches generate solutions that are dependent on the simulation model that they were trained upon. Unfortunately, this implies that results

vary with simulation software, and typically require additional incremental learning when transferred to real robotic systems to adapt for unmodelled effects. In 2021, Algohodhaifi and Lakshmanan published a review of current simulation methods for autonomous vehicles [203]. Open-source (OS) robotics simulators have gained traction among aspiring roboticist for their cost-effectiveness, flexibility, and accessibility. These traits make OS robotics simulators particularly beneficial for academic research. The wide-spread adoption of OS robotics simulation presents a shared framework to collaboratively prototype new algorithms, models, and controllers [204]. Among the current options of OS robotics simulators, Gazebo is the software of choice used in the thesis for flight simulation.

Cook et al. present a useful heuristic to compare robotics simulator software [205]. The approach evaluates each simulator against the following set of criteria: Physical fidelity, programmatic interface, sensor modelling, adequate feature documentation, and previous adoption in research. As an additional criteria, the level of activity by online development communities for the software will also be considered. Therefore, this discussion will exclude simulator tools that are developmentally defunct, e.g. MORSE [206]. The most commonly encountered OS robotics simulator is Gazebo, as it has existed for the longest time among comparable tools, has a large and active community, is highly customizable with ample plugin support, and has undergone significant integration efforts with the Robot Operating System (ROS) framework [207–210]. Riveria et al. present a comprehensive, insightful and informative overview of the usage of modern Gazebo applied to the modelling of ground vehicles in [211]. In my opinion, this work is the best single resource one could consult to gain a structural overview of Gazebo, while also clearly differentiating the functions of ROS from Gazebo. One should contrast the major alternative OS robotics simulators with Gazebo to form an understanding of the current ecosystem. V-REP, which recently has been rebranded to CopelliaSim, is a relatively less popular option compared to Gazebo [212]. While V-REP offers a user-friendly interface and simpler programming techniques, it lacks physical fidelity, with limited development and documentation due to a smaller community [201]. Webots is another popular OS robotics simulation tool that is comparable in user adoption with V-REP, and offers high physical fidelity, an accessible user interface, and a specific design focus on mobile robotics, yet has lower simulation extensibility and less active developer community than Gazebo [213, 214]. While Gazebo is fairly feature rich, the software does not extend well to the creation of large scale and visually rich environments that are closer to the real world. Thus, Gazebo lags in recent advancements for physics-based rendering techniques. Recently, new software tools that leverage modern 3D graphics engines have seen limited use as platforms for dynamical simulation. AirSim is an open-source simulator developed by Microsoft, tailored to autonomous vehicle research [215, 216]. This software is growing in popularity as it offers

high visual fidelity, wide support of multiple programming languages, and has compatibility with open-sourced physics models. However AirSim's overall development and open-source community remain in their infancy, hence have limited application in previous research use. Recent developer efforts on AirSim have focused on implementing the open-source Unity-3D engine as an alternative to the standard and default closed-source Unreal Engine. This represents a promising path to alleviate world scaling concerns that limit current Gazebo implementations [217].

Gazebo is widely used among roboticist for its compatibility with various robot systems and environments. Common applications include autonomous wheeled robots, stationary manipulators, and biomimetic locomotive systems [214, 218–222]. Qian et al. design and simulate a serial link robotic arm with 3 finger grasper using Gazebo in [219]. Notably, the authors' state their usual design process was reduced from several months to less than two, as Gazebo allowed for their software and hardware design to be iterated upon in parallel. Agüero et al. detail the DARPA Virtual Robotics Challenge that takes place within pre-made Gazebo scenarios in [220]. The challenge simulates real-time disaster response for a 28 DoF humanoid robot as it navigates through a series of challenges including driving an electric utility vehicle, walking through different regions with obstacles, and at different points manipulator grasping problems to complete a task in the world. Lei and Ming implement a Q-learning controller training scenario in Gazebo in [221]. The differential drive robot model is trained to perform obstacle avoidance manoeuvring from simulated image depth data. Gazebo accelerates training by enabling automated episode resets. Li and Xiong use Gazebo to simulate a nonlinear MPC for a mobile robot manipulator arm in [222]. Gazebo allows the authors' to generate exact scenario comparisons among different object avoidance schemes. The nonlinear adaptive MPC algorithm they introduce is compared against a potential field method and a rapid-exploration random tree method. Recent research efforts have equipped Gazebo simulation with rudimentary models for of fluid dynamics phenomena to enable the study of water-borne and air-borne drones. Hydrodynamic and hydrostatic effects have been implemented for the study of unmanned underwater vehicles in [223]. The simulation of flying rotorcraft, fixed wings, and airships have also been enabled by simple airfoil and buoyancy model plugins to Gazebo. Imanberdiyev et al. simulate the autonomous navigation of a quad-rotor UAV in [224]. A high-level navigation controller within the simulation loop uses online reinforcement learning to adapt exploration strategies for the simulated physical environment in real-time. Survana et al. simulate a lighter than air autonomous airship model that implements a buoyancy physics model to determine aerostatic lift [225]. Ivanovic et al. use Gazebo to simulate a quad-rotor UAV with a top mounted manipulator arm for indoor agriculture flight scenarios [226]. The UAV is automated to move the manipulator end-effector along a

predefined trajectory corresponding to a regular and repeated camera sensor imaging path, used for fruit yield estimation. Nelissen develops a schematic for a morphing flapping wing aircraft model within Gazebo in [227]. Although the model was not implemented, this work is notable for it provides a highly detailed description for aerodynamic model creation, and includes the idea of using multi-section wings to approximate nonlinear lift models with the basic Gazebo aerodynamics plugin. Pertinent to these efforts is the implementation of hardware-in-the-loop and software-in-the-loop architectures that allow for real-time experimental flight controllers to be validated within simulation [228, 229]. Software in the loop architectures are powerful for bridging the gap between simulated flight control and real flights in a straightforward way. For instance, the remarkable real-life experiment involving the simultaneous flight coordination of a 50 fixed-wing aircraft swarm, as reported by Chung, Davis et al., utilized an intermediate software-in-the-loop simulation step to validate the coordination of the flight controllers [230, 231]. In summary, the significant research efforts undertaken that make use of open-source mobile robotics simulation provide valuable technical insights, and should be leveraged to facilitate a flight simulation methodology to support the development of weight-shift aircraft flight controls.

2.8 Summary of literature review and its connections to the thesis project

The literature review focuses on the body of research relevant to different aspects of the robotic weight-shift flight control that is central to the thesis work. A comprehensive review of dynamics modelling and control research for WSC flexible wing aircraft provides insight for the nonlinear dynamics and modelling uncertainty that characterize flight control challenges for this class of aircraft. While modern model-based and adaptive model-based control techniques are commonly applied to aircraft control problems, they are found to be difficult to implement under conditions where the system model is uncertain and stability is sensitive to modelling error, as is the case for WSC aircraft. Recent developments highlight the effectiveness of model-free techniques that provide robust control solutions for challenging nonlinear dynamic problems without needing detailed models of the system's dynamics. Of specific relevance to the thesis are model-free adaptive critics based ADP techniques that overcome optimal control challenges, such as trajectory tracking, by employing online reinforcement learning.

The robotic piloting system and flight controls presented in the thesis are realized through experimental design of CDPRs and open-source robotic simulation, prior to real flight test. The benefits and challenges posed by cable-driven parallel robots are better understood through the review of different parallel manipulator designs that have been studied in literature. Specific attention is placed on the control problem for CDPRs, as the challenge of coordinating the action for the parallel actuators to realize control objectives is notable and remains an area of active research. The literature on open-source robotics simulators emphasized the critical benefit provided by simulation platforms when validating dynamics models or control systems under development. The applications for mobile robotic problems that make use of Gazebo highlights the important contribution of simulation methodology to mitigating risks prior to real-world testing.

To summarize, this literature review provides a foundation for the research context of the thesis. The literature reference summary in Table 2.1 offers a consolidated overview of the key publications that have been consulted throughout the thesis work. It groups the referenced works according to their main contributions, with the aim that it serves as a guide for quick reference look-up. The literature references have directly informed the development of automatic flight control systems for weight-shift aircraft presented within the papers.

Table 2.1: Summary of key literature references.

References	**Key Contributions**	**Relevance to Thesis Work**
[6, 8, 18, 141, 150, 153]	Historical and foundational dynamics modelling and control of WSC aircraft.	Reviews foundational works and provides a basis for understanding WSC aircraft dynamics, contrasted with conventional fixed-wing aircraft.
[19, 20, 142, 144, 147]	Experimental and theoretical developments in flexible wing WSC aircraft.	Empirical studies providing underpinnings for realizing dynamics models specific to WSC aircraft.
[22–25, 27]	Advanced applications of model-free learning methods in controlling WSC aircraft.	Illustrates advancements in model-free control techniques relevant to handling the inherent complexity of WSC aircraft dynamics.
[159–161, 163,173,175]	Mechanics, theory, and applications for CDPR.	Supports the implementation of CDPRs in developing robotic systems for WSC aircraft, outlining mechanical theories and applications crucial for the thesis.
[168, 177]	Solutions and studies focused on dynamic control using CDPRs.	Provides context on dynamic solutions for CDPRs that operate on a force input level, which are essential to weight-shift actions.
[162, 165, 172]	Comprehensive reviews and practical applications of CDPRs.	Discusses broader research and applications of CDPRs, informing the thesis on design and practical application considerations.
[76, 82, 83, 88, 89, 94]	Fundamental theory on model-free learning control methods within an ADP framework.	Provides a theoretical foundation for the control methodology employed in developing model-free control for CDPRs in WSC aircraft.
[181, 187]	Optimization in CDPR control.	Highlights approaches for online optimization to improve CDPR control, relevant to augmenting conventional methods.
[195, 197]	Neural network-based adaptive learning for CDPR control.	Provides contrast to proposed methods, showcasing adaptive learning performed online in real-time for CDPR systems.
[62, 98, 106, 114, 120]	Theoretical basis for actor-critic approach applied for optimal control.	Supports the methodological approach selected for the real-time implementation studied in the experimental results of the work.
[207, 211, 226,228,229]	Key literature on the use of Gazebo for robotics and aircraft simulation.	Supports the development of the dynamics model and simulation environment for the WSC aircraft in this thesis.

Chapter 3

An Online Reinforcement Learning Wing-Tracking Mechanism for Flexible Wing Aircraft [1]

3.1 Foreword

This chapter presents the first implementation of a candidate parallel actuation scheme for the flight control of a flexible wing weight-shift aircraft. It is adapted from the paper titled "An Online Reinforcement Learning Wing-Tracking Mechanism for Flexible Wing Aircraft", that was published in the proceedings of the 2019 IEEE International Symposium of Robotic and Sensors Environments (ROSE), held on June 17-18th of 2019, in Ottawa, Ontario, Canada. A copy of the original paper is included in Appendix A.1.

Its primary contribution is a weight-shift flight controller based on an adaptive model-free approach that is applied to the kinematic model for a candidate parallel manipulator. The solution is shown in simulation to stabilize the flight dynamics for the Ochi hang-glider model [20], without requiring any knowledge of the aircraft's dynamics. There were two objectives that motivated this work: First, the study centered on incorporating the kinematics of a candidate weight-shift actuation design with a dynamics model for flexible wing weight-shift aircraft. Second, the study aimed to identify a design for a model-free controller that would stabilize the simulated aircraft's flight. The approach was designed

to accommodate uncertain modelling parameters that may dynamically vary during flight. Using online adaptation to adjust the control input based on real-time sensor feedback, the system manages the modelling uncertainties.

This work was motivated by the industry partner's development progress with weight-shift flight actuation schemes. As of the beginning of the fall term of 2018, Romaeris had undertaken a program of experimental flight tests that relied on different wing deformation control schemes, where intentional aeroelastic deflection to the flexible wing's leading edge members, keel, and wing tips were used to induce flight attitude adjustments. It was found that these methods, while functional, provided limited flight control authority. In parallel to the wing deformation control schemes, the industry partner had explored preliminary design concepts of weight-shift actuation schemes that manipulate the wing attitude through control bar inputs, similar to the conventional human-powered flight control method. It remained uncertain if a viable weight-shift actuator design could be realized given the strict engineering constraints on mass, workspace volume, and energy consumption presented by manned flight test operations, and no study had been performed to advance the concepts to an experimental prototype for use in a flight test.

During the active wing deformation control flight test, the industry partner had recorded the 3-axis force input applied by the pilot on the control bar using a proprietary force transducer design. This design and the data remain trade secrets. I pursued a self-led engineering study to characterize attitude control via weight-shifting inputs using the force transducer and test flight sensor data that was provided to me by the partner. With this, I was able to identify bounds on the power and velocity requirements for a hypothetical weight-shift actuation design. Combining this with the test flight sensor data determined that a conventional weight-shift input could be realistically achieved, adhering to the engineering constraints by employing a parallel manipulator design.

This result led to the study of a candidate weight-shift actuation design that featured a pair of parallel linear actuators driven by servo motors, presented in Figure 3.1. I created an empirical model for the forward kinematics of the wing's motion, which is driven by the displacement of the parallel linear actuators. This model was based on experimental data gathered from a proprietary mechanical design under development by the industry partner at the time. The kinematic model was combined with a model-free control method based upon online reinforcement learning. This adaptive control solution stabilized the flight of the flexible wing weight-shift aircraft model without any knowledge of the aircraft's dynamics.

The chapter is organized as follows: Section 3.2 presents briefly the basic operation and model of the flexible wing aircraft system. The combined control structure is detailed out in Section 3.3. The optimal control problem and the associated model-free adaptive learning process are introduced in Section 3.4. The neural network implementation for the online reinforcement learning approach is provided in Section 3.5. Section 3.6 shows the simulation results.

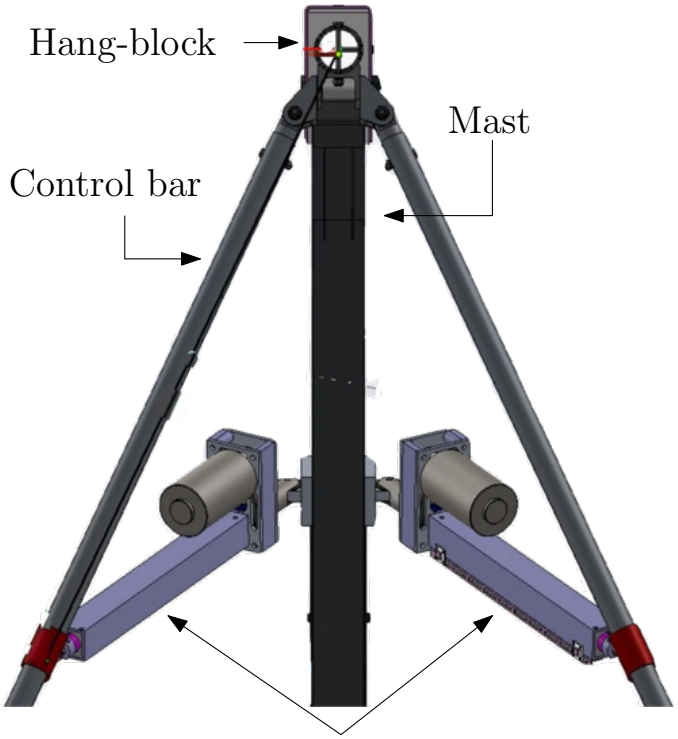


Figure 3.1: CAD model of parallel rigid linear actuator concept considered by the industry partner for weight-shift control.

3.2 Weight-shift manipulator model

The two-body system is attached by a joint that connects the mast of the fuselage (otherwise known as the *chassis*) and the keel tube of the wing, called the hang-block. The hang-block is a mechanical joint which allows for two degrees of motion relative to each

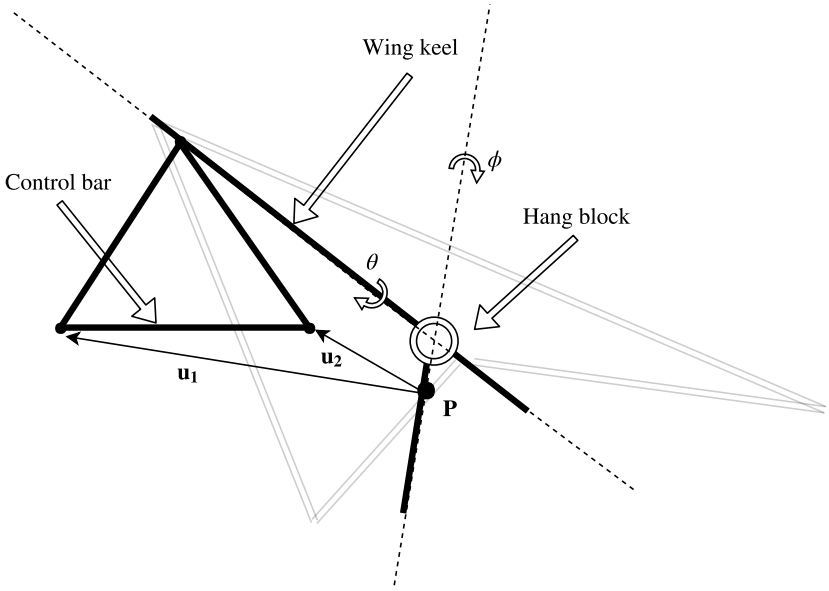


Figure 3.2: Schematic of the wing-fuselage system, connected by a hang-block joint with 2 DoF, ϕ pitch and θ roll, that links the wing-keel to the mast that itself is rigidly attached to the fuselage (chassis). The two attached to the mast at P linear actuators push the wing via the control bar.

body; the wing system may roll freely about the longitudinal axis of the keel bar, and the keel bar may pitch about the axis of the mast. The pilot, who is positioned within the fuselage, modifies the wing's orientation with respect to the fuselage by manipulating the control bar that is rigidly attached to the wing. To modify the vehicle's orientation, the pilot applies force to the control bar to move the wing with respect to the fuselage. By the principle of weight-shift, this in turn modifies the lift profile of the wing, resulting in a new orientation of the wing with respect to the fuselage. For the system considered in this work, the following weight-shift kinematic model is presented with the schematic shown in Figure 3.2.

The orientation of the wing with respect to the fuselage chassis is represented by the pair of rotation angles, θ and ϕ corresponding to roll and pitch of the wing. The set point of the control bar corresponds to the resting orientation of the wing in stable steady altitude, steady speed flight. The actuated motion of the control bar from the set point is represented by the pair of position vectors $\mathbf{u_1}$ and $\mathbf{u_2}$ which describe the distance between the hand grips at the two ends of the control bar, to the pilot represented by a point $\mathbf{P}$ along the

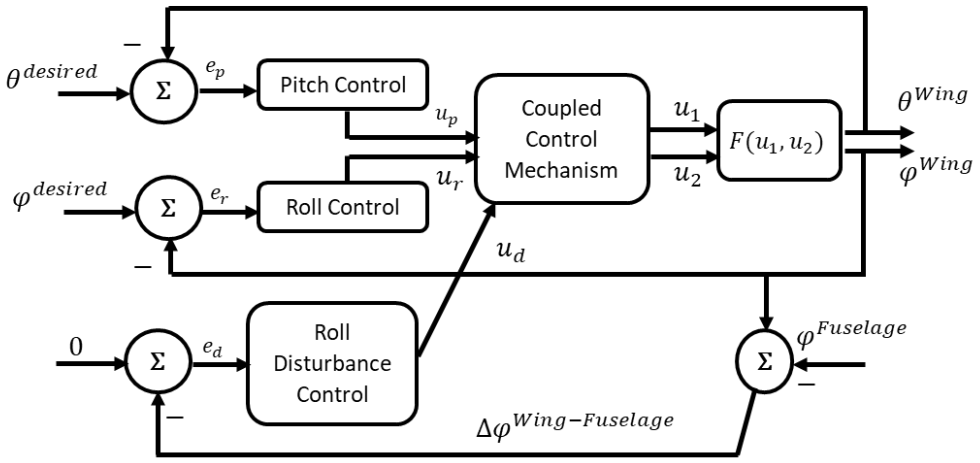


Figure 3.3: Coupled Pitch-Roll Control Mechanism with Wing-Fuselage Roll Disturbances.

mast. The geometric relationship between the orientation of the wing with respect to the fuselage, and the change in position of the control bar is captured by expression

$$(\theta, \phi) = F(u_1, u_2),$$

which maps the wing's orientation ϕ, θ to the pair of distance vectors, $\mathbf{u_1}$ and $\mathbf{u_2}$. To represent any mechanical force disturbances that may act to move the fuselage or wing during flight, an additional noise term is added to the roll component of wing motion, θ^{dist}, and is modelled as a normally distributed random variable with zero mean and variance of $4°$.

3.3 Controller design

This section explains the mechanism of the combined navigation control scheme. As explained earlier, the navigation of the flexible wing system follows a weight-shift mechanism where the center of gravity of the fuselage changes with respect to that of the pilot to achieve the desired maneuvers. Later on, the adaptive learning solution approach is detailed out along with the neural network implementation.

The objective of the control problem is to take into account the pitch and roll orientations of the wing to track certain desired trajectory commands using two actuation systems

with actuation lengths u_1 and u_2 (i.e., the control mechanism will decide two-optimal actuation lengths). The developed mechanism employs two linear actuators in order to induce the required trajectory tracking responses. The combined control mechanism is shown in Figure 3.3, and it depends on a concurrent functioning three-tracking control loops as follows:

- **First loop:** *(i.e., pitch controller)* decides the symmetric actuation lengths u_p to achieve the desired pitching adjustment.
- **Second loop:** *(i.e., roll controller)* selects the asymmetric actuation lengths to allow the desired rolling maneuver. The control signal u_r expresses the desired asymmetry in the actuation lengths to achieve the desired roll adjustments.
- **Third loop:** *(i.e., roll disturbance controller)* considers the roll disturbances created by the relative fuselage-wing discrepancy while achieving the required maneuvers. The regulation signal u_d denotes the required asymmetry in the actuators lengths to suppress the discrepancies between the roll angles of the wing and fuselage and reject other noise that can persist in turbulent flight conditions.

The control mechanism is designed to eliminate the tracking errors (i.e., e_p is the error between the desired pitch orientation $\theta^{desired}$ and the measured pitch orientation θ^{Wing}, e_r is the error between the desired wing roll orientation $\phi^{desired}$ and the actual wing roll orientation ϕ^{Wing}, and e_d is the error due to wing-fuselage disturbances). These tracking control loops employ an adaptive learning approach which utilizes the current and previous errors like PID controllers. Each roll and pitch tracking mechanism uses an adaptive learning algorithm to decide the optimal control gains. The control laws for the tracking error control loops are given as

$$\begin{aligned} u_p(k) &= u_p(k-1) + \Omega_{P1}\, e_p(k) + \Omega_{P2}\, e_p(k-1) + \Omega_{P2}\, e_p(k-2), \\ u_r(k) &= u_r(k-1) + \Omega_{R1}\, e_r(k) + \Omega_{R2}\, e_r(k-1) + \Omega_{R2}\, e_r(k-2), \\ u_d(k) &= u_d(k-1) + \Omega_{R1}\, e_d(k) + \Omega_{R2}\, e_d(k-1) + \Omega_{R2}\, e_d(k-2), \end{aligned} \tag{3.1}$$

where Ω_{P1}, Ω_{P2}, and Ω_{R1}, Ω_{R2} are the longitudinal and lateral tracking control gains that are determined by the adaptive learning process that will be defined later on.

Note that the third control loop (i.e., disturbances regulation loop) uses the same control gains obtained for the roll tracking mechanism, as its corrections use the same control structure introduced by roll mechanism (i.e., the second tracking control loop). This is because

the most significant modelled disturbances affect the roll response disproportionately [30]. The coupled control mechanism block shown in Figure 3.3 makes a logic combination of the separate selected variations in the actuation lengths u_p, u_r, and u_d. This unit builds the logic of how the actuation lengths u_1 and u_2 are inter-chosen to achieve the desired maneuvers. The resulting combined control laws are expressed as follows:

$$u_1 = u_p - (u_r + u_d)/2,$$
$$u_2 = u_p + (u_r + u_d)/2.$$

Each control law decides the symmetric (pitching) and asymmetric (rolling) fractions of the actuation movements. It is noted that the asymmetric actuation movements (u_r and u_d) cause partial pitch deviations which is corrected during the flight by the pitch control loop, which is done in a dynamic operation. Furthermore, this unit considers the minimum and maximum constraints on the allowed actuation lengths. This means that if one actuator is hitting the bounds, the over-limit actuation is compensated for by the other actuator due to the symmetric inter-distribution capability of the actuators. The parallel actuators u_1 and u_2 are corrected to a point in the allowed actuation length range. The next section lays out the mathematical foundation to build the adaptive learning control mechanism.

3.4 The optimal control problem

This section explains the model-free approximate dynamic programming approach and develops an adaptive learning approach based on an online value iteration process. First, the temporal difference equation, i.e. Bellman equation, is stated and then the optimality principle [42, 62] is applied in order to derive the optimality conditions. The proposed mechanism works to minimize the tracking errors (i.e., e_p, e_r and e_d) to achieve the desired mixed longitudinal-lateral maneuvers. Thus, a simple diagram for a simple tracking unit is shown in Figure 3.4. The signals $y^{desired}$ and $y^{measured}$ designate the desired and measured signals respectively, $e(k)$ is the tracking error signal, and $u(k)$ is the selected control signal for the dynamical system.

Herein a tracking mechanism that is similar to the basic operation of a PID controller will be introduced. This is done using an online model-free adaptive learning mechanism. This mechanism does not need the aerodynamic model of the underlying flexible wing aircraft. Later on, this unit is integrated into the combined control mechanism.

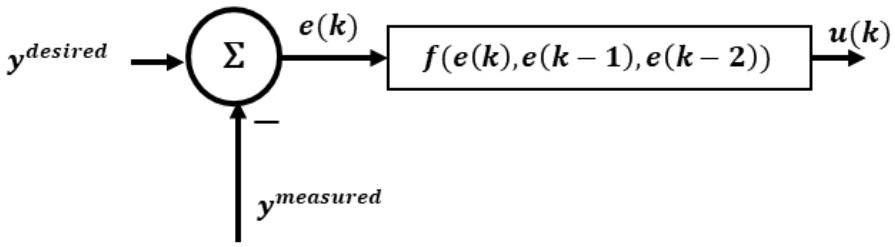


Figure 3.4: The tracking control unit, time delayed error signals are used for PID-like input data structure.

In this case, the main states of the adaptive learning approach are the tracking error variables (i.e., $e(k), e(k-1), e(k-2), \forall k$, where k is the time-index). The objective of the optimal control problem for each tracking unit is to select the optimal control strategy $u^o(k), \forall k$ in order to minimize a quadratic convex cost function or simply fulfill the tracking $e(k), \forall k \to 0$ objectives. Thus, the objective cost function is given by

$$C\left(\boldsymbol{E}(\ell), u(\ell)\right) = \frac{1}{2}\left(\boldsymbol{E}^T(\ell)\,\boldsymbol{Q}\,\boldsymbol{E}(\ell) + u^T(\ell)\,R\,u(\ell)\right),$$

where $E(\ell)$ is a vector of the errors so that $E(\ell) = [e^T(\ell)\ \ e^T(\ell-1)\ \ e^T(\ell-2)]^T$, $R > 0 \in R^{1\times 1}$ and $\boldsymbol{Q} \geq 0 \in R^{3\times 3}$ are symmetric weighting matrices. $\boldsymbol{Q}$ is selected by the designer to weigh the relative importance of the states for optimization objective.

This process is repeated for each tracking control loop. The utility function $C(\dots)$ is used to form the following performance index P

$$P = \sum_{\ell=0}^{\infty} C\left(\boldsymbol{E}(\ell), u(\ell)\right),$$

The control strategy $u(\ell), \forall \ell$ is selected in order to minimize the control effort and hence the performance index. The necessary optimality conditions are found by applying Bellman's optimality principle to define the Bellman equation [62, 82]. For simplicity $\boldsymbol{E}(\ell), u(\ell)$ are referred to as E, u for the same time index. A quadratic value function solution form is used such that

$$S(\boldsymbol{E}, u) = \frac{1}{2}[\boldsymbol{E}^T\ \ u^T]\ \boldsymbol{\omega}\ \begin{bmatrix} \boldsymbol{E} \\ u \end{bmatrix}$$

where $\boldsymbol{\omega} = \begin{bmatrix} \boldsymbol{\omega}_{EE} & \boldsymbol{\omega}_{Eu} \\ \boldsymbol{\omega}_{uE} & \boldsymbol{\omega}_{uu} \end{bmatrix}$, $\omega_{\boldsymbol{Eu}}$, $\omega_{\boldsymbol{uE}}$, $\omega_{\boldsymbol{EE}}$, and $\omega_{\boldsymbol{uu}}$ are sub-matrix components of the matrix $\boldsymbol{\omega}$.

The quadratic value function $S(\boldsymbol{E}, u)$ defined in (3.2) quantifies the expected cumulative cost associated with the current state $\boldsymbol{E}$ and control input u. Bellman's optimality principle asserts that the value of the current state-action pair is equal to the immediate cost $C(\boldsymbol{E}, u)$ plus the value of the subsequent state-action pair $S(\boldsymbol{E}(\ell+1), u(\ell+1))$. The performance index P and the solution form S are used to formulate the temporal difference equation, which is equivalent to Bellman equation, so that

$$S(\boldsymbol{E}(\ell), u(\ell)) = C\left(\boldsymbol{E}(\ell), u(\ell)\right) + S(\boldsymbol{E}(\ell+1), u(\ell+1)),$$

The optimality conditions are obtained using Bellman's optimality principle so that

$$u^o = \underset{u}{\operatorname{argmin}} \left(S(\boldsymbol{E}, u)\right).$$

Then $\boldsymbol{\omega}_{uu} u + \boldsymbol{\omega}_{u\boldsymbol{E}} \boldsymbol{E} = 0$. Therefore, the model-free control strategy is given by

$$u^o = -\left[\boldsymbol{\omega}_{uu}^{-1} \boldsymbol{\omega}_{u\boldsymbol{E}}\right] \boldsymbol{E}. \tag{3.2}$$

Hence the Bellman equation takes the form

$$S^o(\boldsymbol{E}(\ell), u^o(\ell)) = C\left(\boldsymbol{E}(\ell), u^o(\ell)\right) + S^o(\boldsymbol{E}(\ell+1), u^o(\ell+1)). \tag{3.3}$$

Solving (3.3) using the optimal control strategy (3.2) guarantees a solution for the underlying optimal control problem (i.e., stabilizing the tracking errors $e(\ell), \forall \ell$).

An adaptive learning process is proposed to solve all the Bellman equation (3.3), i.e. Bellman equations for all the tracking control loops, using the optimal control strategies (3.2). Value iteration is an adaptive learning process that is concerned with solving the temporal difference equation or the underlying Bellman equation (3.3) [79, 83, 88, 92]. This process is composed of two main steps; First, value function evaluation, and second the control strategy improvement.

The value iteration algorithm is arranged as shown with Algorithm 1. This algorithm is implemented in real-time and it employs a model-free control strategy. It runs concurrently for the different scheduled control loops as will be explained later on. The following section shows the implementation of this reinforcement learning algorithm using the neural network approximations.

Algorithm 1 Value Iteration Solution

1: Initialize the solution $S^0(\dots)$ and u^0.
2: Evaluate the new solution matrix $\omega^{r+1}, \forall r$ following

$$S^{r+1}(\boldsymbol{E}(\ell), u(\ell)) = C^r\left(\boldsymbol{E}(\ell), u(\ell)\right) + S^r(\boldsymbol{E}(\ell+1), u(\ell+1)) \tag{3.4}$$

where r is the value iteration solution index.
3: Evaluate improvement in the control strategy following

$$u^{r+1} = -\left[\boldsymbol{\omega}_{uu}^{-1}\boldsymbol{\omega}_{u\boldsymbol{E}}\right]^{r+1} \quad \boldsymbol{E}. \tag{3.5}$$

4: Continue with the process till the differences $\Delta\boldsymbol{\omega}^{r-1,r} = \boldsymbol{\omega}^r - \boldsymbol{\omega}^{r-1}$ converge (i.e., the errors are less than a threshold) for a predefined number of iterations N.

3.5 Neural network solution mechanism

This section explains how the adaptive learning mechanism proposed by Algorithm 1 can be implemented using the adaptive critics or the neural network structures [92, 99, 100, 133,137]. The solution value function $S(\dots)$ and the associated control strategy $u(\dots)$ are approximated using separate neural network structures. This process enables capturing the optimality in the dynamic learning environment, where the taken strategies are rewarded or penalized based on the accumulated cost values. The adaptation of the weights is done in real-time using simple gradient descent approach and linear quadratic cost functions are employed.

The solution form is defined as a value function $S(\dots)$, and is approximated by the following neural network structure

$$\hat{S}(\boldsymbol{E}, \hat{u}) = \frac{1}{2}[\boldsymbol{E}^T \quad \hat{u}^T] \ \boldsymbol{W}^T \ \begin{bmatrix} \boldsymbol{E} \\ \hat{u} \end{bmatrix},$$

where W^T is the matrix of approximation weights for the solution value function S.

Consequently, the associated optimal control strategy $u(\dots)$ is approximated using the following neural network structure

$$\hat{u} = \boldsymbol{\Omega}^T\boldsymbol{E},$$

where Ω^T are the approximation weights for the optimal control strategy.

The solution value function's approximation error ε_W is given by

$$\varepsilon_W = \left(\hat{S}(\boldsymbol{E}, \hat{u}) - \tilde{S}\right),$$

where the value $\tilde{S}$ is given by

$$\tilde{S} = C\left(\boldsymbol{E}(\ell), \hat{u}(\ell)\right) + S(\boldsymbol{E}(\ell+1), \hat{u}(\ell+1)),$$

Hence, the update rule for the approximation weights W^T is given by

$$\boldsymbol{W}^{(r+1)T} = \boldsymbol{W}^{(r)T} - \eta \left(\left(\hat{S}(\boldsymbol{E}, \hat{u}) - \tilde{S} \right) \begin{bmatrix} \boldsymbol{E} \\ \hat{u} \end{bmatrix} \begin{bmatrix} \boldsymbol{E}^T & \hat{u}^T \end{bmatrix} \right)^{(r)}, \tag{3.6}$$

where $0 < \eta \leq$ is a learning rate. The gradient descent approach is used due to the simplicity of the linear quadratic cost function $C(\dots)$ which enables this direct formalization. In a similar fashion, the approximation error for the control strategy is given by $\varepsilon_u = (\hat{u} - \tilde{u})$ where

$$\tilde{\boldsymbol{u}} = -\left[\boldsymbol{\Omega}_{\hat{u}\hat{u}}^{-1} \boldsymbol{\Omega}_{\hat{u}\boldsymbol{E}}\right] \quad \boldsymbol{E}.$$

Similarly, the adaptation rule for the control strategy approximation weights follows

$$\boldsymbol{\Omega}^{(r+1)T} = \boldsymbol{\Omega}^{(r)T} - \mu \left((\hat{u} - \tilde{\boldsymbol{u}})\, \boldsymbol{E}^T \right)^{(r)}, \tag{3.7}$$

where μ is a learning rate.

A full online neural network solution algorithm can be arranged as shown with Algorithm 2.

Algorithm 2 Neural Network Solution

1: Initialize the weights $W^0(\dots)$ and Ω^0.
2: Start with arbitrary small error values $e(-2), e(-1)$ and $e(0)$.
3: Calculate the approximation $\hat{u}$.
4: Compute $\hat{\boldsymbol{S}}$.
5: Evaluate the measurement $e(r)$ and arrange the errors' vector $\boldsymbol{E}$.
6: Calculate the approximation $\hat{u}$ and $\hat{\boldsymbol{S}}$ at iteration r.
7: Find $\tilde{\boldsymbol{S}}$.
8: Evaluate the new solution value approximation weights using (3.6).
9: Decide the improvement in the control strategy using (3.7).
10: Continue with the process till the differences $\Delta \boldsymbol{W}^{r-1,r} = \boldsymbol{W}^r - \boldsymbol{W}^{r-1}, \forall r$ and $\Delta \boldsymbol{\Omega}^{r-1,r} = \boldsymbol{\Omega}^r - \boldsymbol{\Omega}^{r-1}, \forall r$ converge (i.e., the errors are less than a threshold) for a predefined number of iterations N.

3.6 Simulation results

In this section, the simulation of the Ochi hang-glider dynamics model is performed [20]. The combined pitch-roll control strategy is employed to test its ability to stabilize the system. Weight-shift action is governed by the parallel manipulator kinematic model function

$F(u_1, u_2)$. The contents of this function is intentionally left obfuscated to protect trade secrets pertaining to the actuator development by the industry partner. First, the adaptive learning algorithm is applied to the online decoupled maneuvers to advise the control gains. This is followed by a planned pitch-roll combined maneuver to show the actuation performance in order to achieve the desired maneuver. The learning rates are selected so that $\mu = \eta = 0.001$.

The simulation is performed for longitudinal maneuvers to track desired pitch angles from -15° to 15° with 5° steps. The adaptations of the control strategy gains are shown in Figure 3.5(a). It can be seen that the optimal control gains converge within relatively few evaluation steps. The corresponding wing orientations as shown in Figure 3.5(c) in reaction to the actuation lengths plotted in Figure 3.5(b) demonstrate the correct performance of the learned controller. The rapid convergence property of the adaptation process is further emphasized by examining the controller's action for alternative lateral maneuvers in the range of -20° to 20° with 5° steps. The adaptations of the controller weights are shown to converge quickly as shown in Figure 3.6(a). The actuation lengths are the same since they are moving symmetrically Figure 3.6(b), to achieve the desired maneuver plotted in Figure 3.6(c). It is worth to note that the asymmetry in the lateral actuation motion induces marginal pitch adjustments. These adjustments are compensated for, automatically, through the pitch control regulation.

The combined trajectory-tracking control mechanism is tested using the predefined three control loops working all together simultaneously, with the following control gains $\Omega_p = [0.0251 \quad -0.0320 \quad -0.0139]$ and $\Omega_R = [0.0088 \quad -0.0319 \quad -0.0102]$. The desired decoupled maneuvers are formed of two turns in each input $\theta^{desired}(r) = -10sin(2\pi r/1500)$ and $\phi^{desired}(r) = 10sin(2\pi r/1500)$ and they happen concurrently. The wing-fuselage disturbances are modelled using stochastic noise from a normal distribution with amplitude $\pm\, 4^\circ$.

Figure 3.7 shows the different actuation lengths provided by the control loops to achieve the desired maneuvers. The direct actuation lengths applied to the flexible wing system are shown in Figure 3.8(a), while the actual performance compared to the desired trajectory is plotted in Figure 3.8(b). These figures emphasize the ability of the combined control mechanism to satisfy the desired tracking objectives with very small lag using the online model-free adaptive learning mechanism.

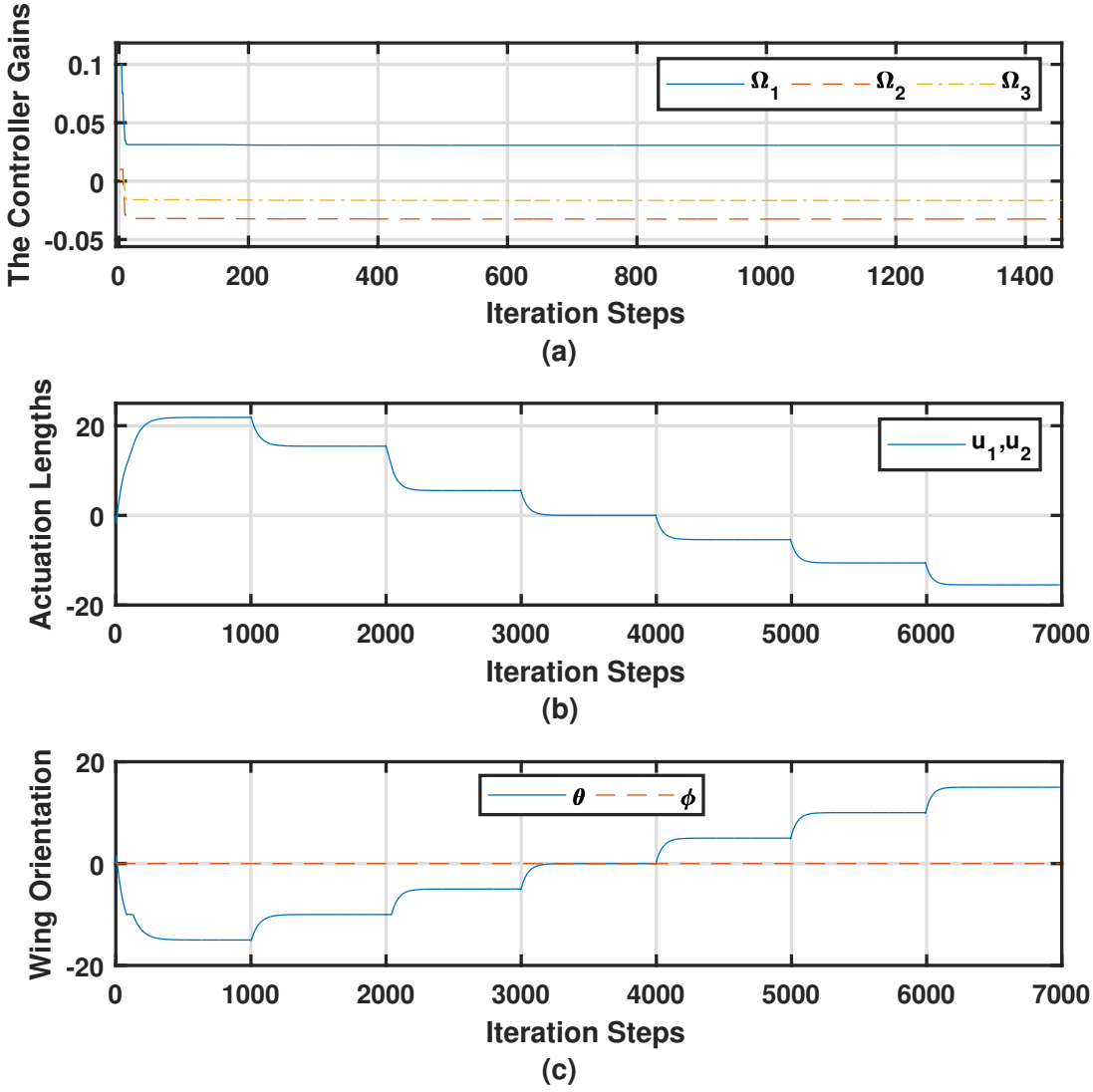


Figure 3.5: Longitudinal motion. (a) The adaptation of the controller's gains. (b) The actuation lengths in response to desired maneuver. (c) The resulting wing orientation.

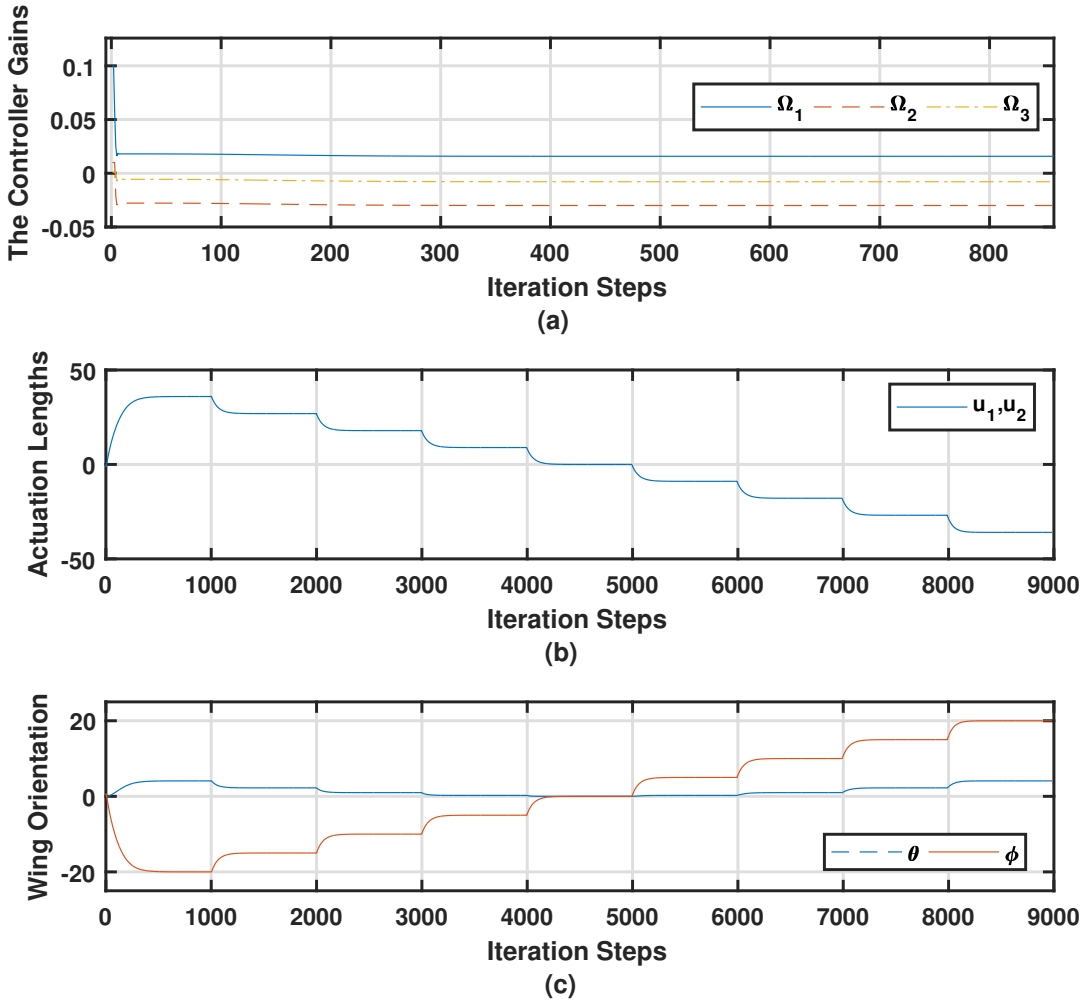


Figure 3.6: Lateral motion. (a) The adaptation of the controller's gains. (b) The actuation lengths in response to desired maneuver. (c) The resulting wing orientation.

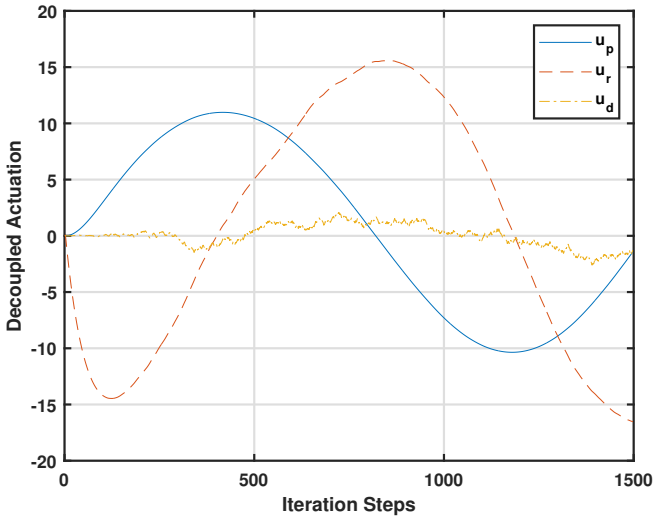


Figure 3.7: The unadjusted actuation lengths by the different controllers.

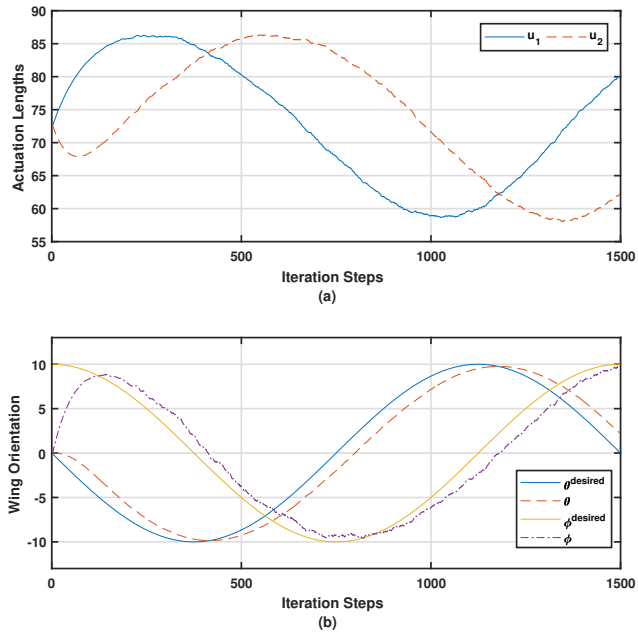


Figure 3.8: Dynamic response of the coupled maneuver. (a) The actuation lengths. (b) The coupled longitudinal-lateral maneuvers.

3.7 Chapter summary

This chapter introduced a model-free trajectory-tracking control mechanism for a flexible wing weight-shift aircraft system. This represents the first time in literature that a simulated weight-shift aircraft incorporates the kinematic model of a candidate robotic manipulator design. The controller consists of three control loops that coordinate the action of the parallel actuators while accounting for the interactions between the fuselage body and wing body systems. An online adaptive learning mechanism based on a value iteration process is employed to tune the control gains of the different control loops by measuring the performance of the systems as they interact with the weight-shift flight. The adaptive controller does not rely on any information about the aerodynamics of the flexible wing or the dynamics of the two-body wing-fuselage system. The solution of the adaptive algorithm is implemented and approximated using separate online neural networks. The simulation results demonstrate the effectiveness of the proposed controller design in different trajectory-tracking simulation cases. The optimal control action manages the multiple pitch-roll tracking and stabilization objectives under weight-shift inputs through performant coordination of the parallel actuators.

Chapter 4

Guidance Mechanism for Flexible Wing Aircraft Using Measurement-Interfaced Machine Learning Platform [2]

4.1 Foreword

This chapter presents the first hardware implementation within this thesis of an adaptive actor-critic online learning controller on real flight controller hardware, used to control a dynamically unstable system using parallel manipulators. It is adapted from the paper titled "Guidance Mechanism for Flexible Wing Aircraft Using Measurement-Interfaced Machine Learning Platform" that was published within the IEEE Transactions on Instrumentation and Measurement journal, in July of 2020. A copy of the original paper is included in Appendix A.2.

This work was motivated by the need to prove that an online learning method could be deployed on computationally constrained flight controller hardware, despite the real-time control requirement that the hardware presents. This process addresses difficulties associated with developing control solutions for partially or fully model-based approaches with uncertain dynamics. The model-free solution demonstrates good trajectory tracking

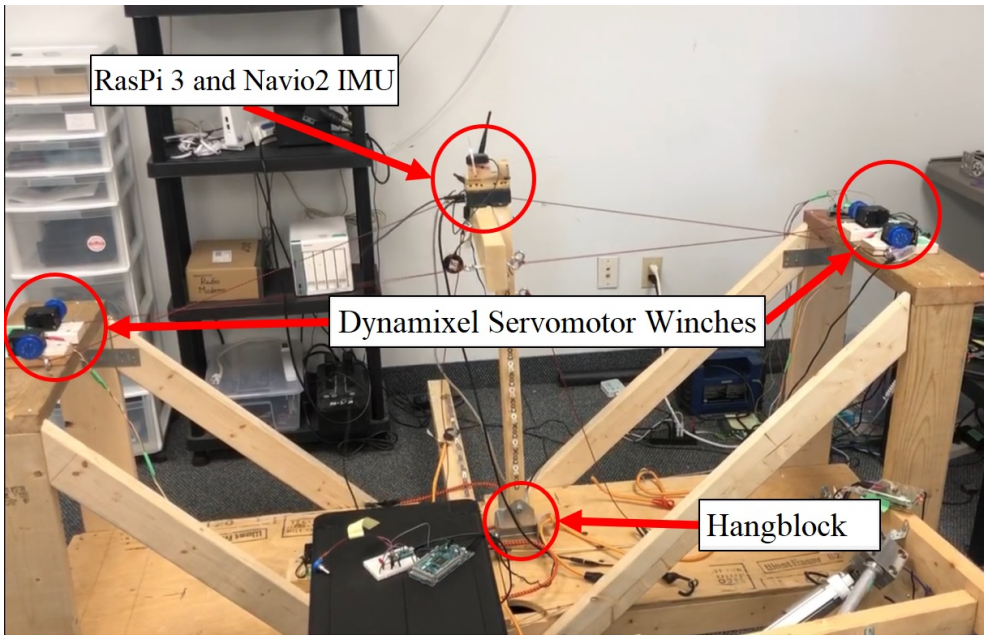


Figure 4.1: Experimental mock-up used to emulate the motion of the fuselage about the wing in ground test. Note that the wing-mast system is oriented upside down; i.e., the top surface of the middle bar, representing the wing, is facing the floor, while the mast likewise sits above it. The figure also shows the Navio2+Raspi IMU and controller, servomotor winches, and hang-block.

performance in the face of large disturbance events. The experimental mock-up is a 2-axis inverted pendulum, with kinematics identical to the hang-block of a weight-shift aircraft, shown in Figure 4.1. The inverted pendulum mock-up is essentially a reduced or simplified version of a weight-shift aircraft's flight control problem. Both the inverted pendulum and the weight-shift aircraft are dynamically unstable systems that require active control inputs to maintain their balance beyond their unstable equilibrium. With both systems, small control inputs lead to significant changes in the system's state, hence both systems are sensitive to disturbances.

This work advances initial research investigations aimed at designing adaptive flight controllers for flexible-wing aircraft using parallel actuators, as introduced in Chapter 3. However, this development relied on an ideal kinematic model of parallel linear actuator displacement. Further, the evaluation of the controllers was limited to their performance based solely on tracking error. In contrast, this paper presents an online learning mech-

anism using real hardware that determines control input based on tracking errors, their time-delayed dynamics, and the dynamical measures of the system state feedback parameters to support the overall stability of the system. To implement the learning algorithms with real hardware, I developed several specific innovations in online reinforcement learning techniques:

- The use of time-delayed signals to form PID-like information structure that are inputs for the adaptive learning units.
- Introduction of a second reinforcement learning agent which acts to augment the stability of the physical mechanism, preventing it from entering unstable regions of its phase space, and must cooperate with the reference error learning tracking unit.
- A variant of the model-free learning process based on real-time value iteration is implemented, where the actor and critic units evolve according to stochastic gradient descent based adaptation laws.
- The actor and critic are assigned different learning rate scales to prevent sudden adverse changes to the control policy that would otherwise be induced by temporarily poor value function approximations.
- A set of gradient directional constraints, where designer-defined vectors modify the value function gradients to steer policy optimization.

The primary contribution of the paper is the online solution to the optimal trajectory tracking problem realized by the introduction of a second adaptive agent that, by observing system's state feedback, acts as a dynamic stability augmentation system. The inspiration for this technique is rooted in the structure of classical offline solutions for optimal trajectory tracking control problems, where a trade-off is necessary between immediate error correction and overall tracking stability. The action of the stability support agent augments the action of the primary error controller to improve overall trajectory tracking performance in an optimal sense. The stability support agent induces a trade-off for short term error correction performance to ensure that the weight-shift system remains near the reference trajectory. The overall performance of the solutions depends on the implicit coordination of the primary error tracking agent and the secondary stability supporting agent. The solution provides optimal trajectory tracking in a model-free way and requires no model of the system dynamics.

The secondary contribution of the paper is the model-free guided search learning technique. The guided search mechanism informs the adaptation of the actor's neural network by weighting the critic neural network's output components based on the dynamic measures of concern identified by the designer. This technique provides an improved rate at which the actor adaptation process yields a viable stable controller, at the expense of a sub-optimal performance as defined by the critic.

The results presented in the paper served to inform the industry partner on the practicality of reinforcement learning methods for the purpose of controls design. The ability of the model-free solution to provide trajectory tracking control that was robust to disturbances and could manage the nonlinear dynamics presented by the weight-shift surrogate system supported further exploration of a weight-shift actuator based piloting system.

The chapter is organized as follows: The operation of flexible-wing aircraft is briefly highlighted in Section 4.2. The measurement scheme and the adopted real-time sensory devices are detailed in Section 4.3. The different control and optimization objectives are presented in Section 4.4. This is followed by the development of the machine learning process and its adaptive critics implementation in Section 4.5. The experimental results are analyzed in Section 4.6 in order to evaluate the validity of the proposed control setup.

4.2 Operation of flexible-wing aircraft

In this section, the basic kinematic relationships concerned with solely the longitudinal motion control of the flexible-wing aircraft are detailed. The two-mass system (i.e., wing and fuselage) attaches the two bodies by a joint known as the hang-block which connects the mast of the fuselage and the keel tube of the wing at a common point known as the hang point. The hang-block is a mechanical joint which allows for two degrees of rotational motion, pitch and roll, relative to each body. The wing system may roll freely about the longitudinal axis of the keel tube, and the keel tube may pitch about the axis which is perpendicular to both the mast and keel tubes. During manned flight, the pilot who is positioned within the fuselage, modifies the wings pitch orientation with respect to the fuselage by pushing or pulling against the control bar that is rigidly attached to the wing. To modify the wing's orientation, the pilot applies a force to the control bar to shift the fuselage with respect to the wing. By the principle of weight-shift, this in turn modifies the balance of forces on the wing, resulting in a new wing orientation with respect to the fuselage.

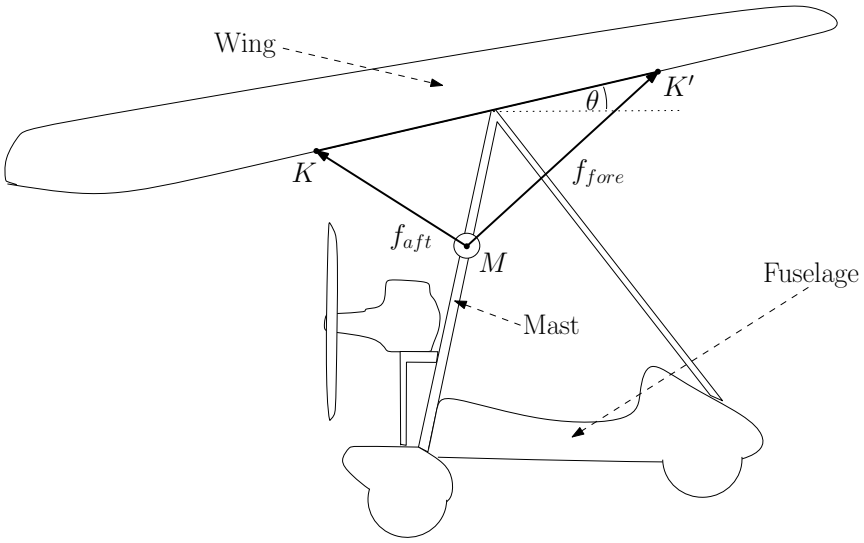


Figure 4.2: Schematic of weight shift kinematics of flexible-wing aircraft with two servo winches acting on wing to affect pitch.

An experimental mock-up is set up to emulate the relative actuation mechanism between the wing and fuselage. The weight-shift kinematics of this mechanism is schematically depicted in Figure 4.2. A pair of servomotor winches are mounted at point M on the mast, and connected to points K and K' on the keel tube. The orientation of the wing pitch with respect to the fuselage is represented by θ. It is important to note that the wing does not rotate along the yaw axis, as it is constrained by the hang point on that axis. The set point of the control bar corresponds to the resting orientation of the wing in stable steady altitude and steady speed flight. Pull-only forces applied to the wing by the winch servomotors are represented by the pair of position vectors f_{fore} and f_{aft}, and act in the free motion axis of pitch perpendicular to the keel tube with unknown angles proportional to $\angle KK'M$ and $\angle K'KM$, respectively. For this work, it is assumed that the wing control in the pitch axis is sufficiently decoupled from the roll axis, such that the servomotor winch actions due to their geometric configuration do not induce the roll motion of the wing.

4.3 Instrumentation and measurement platform

The experimental mock-up emulating the relative motion between the fuselage and the wing is shown in Figure 4.1. A block diagram of how the system's interconnected blocks are interfaced is revealed in Figure 4.3. Efforts were made to ensure each hardware device and supporting software is either open-source or has publicly available documentation. An Emlid Navio2 [232] board attached to a Raspberry Pi 3 [233] is selected as the controller's computational unit due its relatively convenient and easy user setup and prevalence as a hardware of choice for open-source flight controller platforms, such as ArduPilot [234] and ROS [235]. Navio2 provides redundant wing orientation estimates through the combination of sensor readings from either the main on-board InvenSense MPU-9250 9-Degree-of-Freedom (DOF) Inertial Measurement Unit (IMU) [236] or the secondary Microelectronics LSM9DS1 9 DOF IMU [237], as well as global positioning system (GPS). Each IMU has a 3-axis magnetometer, 3-axis gyroscope, and 3-axis accelerometer. The IMU signals are filtered by a programmable digital low pass filter, which is then fed to a portable mini-computer (Raspberry Pi 3 model B). The unit has a 1.2GHz 64-bit quad-core ARMv8 CPU, 1GB of RAM and a Cortex-M3 co-processor.

Angular position measurements of the wing orientation were sampled at 20 Hz, a rate far below the maximum of 8 kHz supported by the dual IMUs. The total root mean square (RMS) noise of the MPU-9250 used for inertial measurement feedback is provided to be $0.1\,^{\circ}/\mathrm{s}$. The measurements data are filtered by a low-pass filter with a cut-off frequency of 250 Hz. The control of the platform was actuated by two Dynamixel XM-430 servomotors [238]. The servomotors receive current input commands at a rate of 20 Hz from the controller embedded within the Raspberry Pi 3. In reference to Figure 4.1, the system is arranged as if the aircraft wing was upside down, where the hang-block and wing are closest to the ground, and the mast sits above it. This setup was chosen for its convenience but it should have no effect on the validity of the experimental results. The mock-up allows for the mast to move relative to the wing. The servomotor winches are rigidly attached to the wing keel, and the attachment points are at opposite points along the mast. The results presented in Section 4.6 are captured from experiments with this mock-up system where two servomotors mounted are directly on the surrogate wing keel. Desired reference position commands are programmed prior to the experiments. The learning algorithm decides the best control policy using IMU observations, in real-time, and transmits current signals, based torque commands, to the servomotors to track the reference signal.

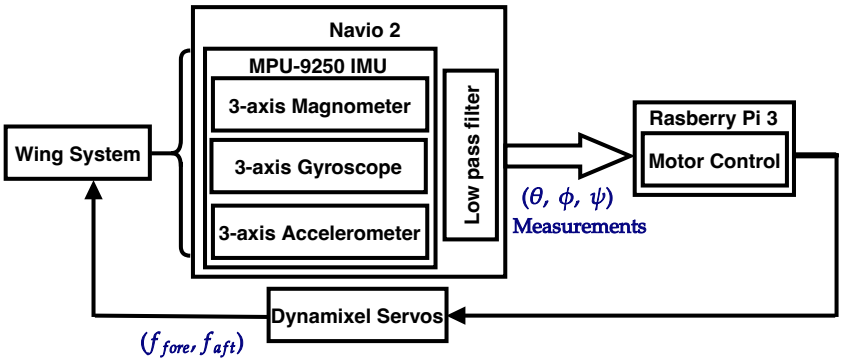


Figure 4.3: Flow schematic of experimental instrumentation.

4.4 The control and optimization problem

In the following section, ideas behind the online machine learning process are developed. Thereafter, the governing temporal difference equations are derived. Further, this section discusses how the different control tasks, i.e., tracking and stabilization, are coordinated simultaneously in real-time.

4.4.1 Optimal control structure

Due to the aircraft's complicated nonlinear dynamics, it is necessary to avoid building a control or computational approach that relies explicitly on existence of an aerodynamic model. As such, any control solution would be better based on robust model-free computational mechanisms.

The wing's pitch control mechanism is realized using two interacting control objectives. The first looks for a control strategy that receives real-time tracking error signals, arranged using a PID-like structure, and provides a control signal that minimizes the tracking error. The second aims for an auxiliary control policy that supports the stability of the overall system during the maneuvers. The full control system is schematically represented in Figure 4.4. Both control objectives are integrated to act together. Each employs a guided search algorithm to constrain policy gradient directions, which is described later on in Section 4.5.

The structure of the tracking error vector $E_\ell \in \mathbb{R}^o$ reflects the design objectives corresponding to the tracking control strategy π_E which in turn decides the tracking control signal $u_\ell^{\pi_E} \in \mathbb{R}^m$. The error vector E_ℓ relies on various dynamic forms of the tracking error signals e_ℓ (ℓ is a time-index). The stabilizing control policy π_X maps the observable key measurements in vector $X_\ell \in \mathbb{R}^n$ to a state feedback control signal $u_\ell^{\pi_X} \in \mathbb{R}^s$ in order to support the system's stability during the navigation process. The overall control $u_\ell^F \in \mathbb{R}^{\max\{s,m\}}$ which is applied to the actuation system, results from combining the dynamical effects from both control policies (i.e., π_E and π_X). The control input signals from the two learning agents, u^{π_E}, u^{π_X}, contain the set of intermediate signals that sum to u^F, the resulting set of control inputs of the parallel actuators responsible for moving the control bar of the flexible-wing aircraft. It is worth to note that in order to generalize the proposed approach, the control signals u^{π_E} and u^{π_X} could have different vector sizes, and the total control signal u^F sums the actuation signals in both adaptive units. In this work, the control signals which are fed to the actuation systems are scalars. This simplifies the form of the collective control signal u^F.

Performance indices $J^{\pi_E}(E_\ell)$ and $J^{\pi_X}(X_\ell)$ are used to asses the usefulness of the attempted policies π_E and π_X, respectively, so that

$$\begin{aligned} J^{\pi_E}(\boldsymbol{E}_\ell) &= \sum_{i=\ell}^{\infty} C^E\left(\boldsymbol{E}_i, \boldsymbol{u}_i^{\pi_E}\right) \\ J^{\pi_X}(\boldsymbol{X}_\ell) &= \sum_{i=\ell}^{\infty} C^X\left(\boldsymbol{X}_i, \boldsymbol{u}_i^{\pi_X}\right), \end{aligned} \tag{4.1}$$

where C^E and C^X are cost functions associated with the performance indices $J^{\pi_E}(\boldsymbol{E}_\ell)$ and $J^{\pi_X}(\boldsymbol{X}_\ell)$, respectively. They are given by

$$\begin{aligned} C^E(\boldsymbol{E}_\ell, \boldsymbol{u}_\ell^{\pi_E}) &= \frac{1}{2}\left(\boldsymbol{E}_\ell^T\, \boldsymbol{Q}^E\, \boldsymbol{E}_\ell + \boldsymbol{u}_\ell^{\pi_E\, T}\, \boldsymbol{R}^E\, \boldsymbol{u}_\ell^{\pi_E}\right) \\ C^X(\boldsymbol{X}_\ell, \boldsymbol{u}_\ell^{\pi_X}) &= \frac{1}{2}\left(\boldsymbol{X}_\ell^T\, \boldsymbol{Q}^X\, \boldsymbol{X}_\ell + \boldsymbol{u}_\ell^{\pi_X\, T}\, \boldsymbol{R}^X\, \boldsymbol{u}_\ell^{\pi_X}\right), \end{aligned}$$

where $\boldsymbol{Q}^E \in \mathbb{R}^{o\times o}$, $\boldsymbol{R}^E \in \mathbb{R}^{m\times m}$, $\boldsymbol{Q}^X \in \mathbb{R}^{n\times n}$, and $\boldsymbol{R}^X \in \mathbb{R}^{s\times s} > 0$ are symmetric positive definite weighting matrices.

The objective functions C^E and C^X have convex forms and quadratic dependencies on the different policies and real-time measurements. These forms enable Lyapunov stability proofs for the underlying temporal difference solutions, as shown in [131, 133].

The design of the optimization problem could vary from the one that is considered herein. This is decided by the designer who judges the different segments of the optimization problem and how they are hierarchically organized. In this development, the design depends on two problem characteristics: the way key measurements are available and calculated for the control design, and the dynamics which the controller needs to consider or regulate during the navigation process.

4.4.2 Mathematical solution framework

The control solution developed herein maps the optimization objectives mentioned above into temporal difference forms using a discrete-time optimal control framework [62]. Quadratic forms of solving value functions $V^{\pi_E}(\boldsymbol{E}_\ell)$ and $V^{\pi_X}(\boldsymbol{X}_\ell)$ are employed to evaluate the quality of the computed control strategies. They are based on the structures of the indices $J^{\pi_E}(\boldsymbol{E}_\ell)$ and $J^{\pi_X}(\boldsymbol{X}_\ell)$ and the associated cost functions C^E and C^X, respectively, such that

$$\begin{aligned} V^{\pi_E}(\boldsymbol{E}_\ell) = \frac{1}{2}\boldsymbol{E}_\ell^T\, \boldsymbol{S}^E\, \boldsymbol{E}_\ell &\equiv J^{\pi_E}(\boldsymbol{E}_\ell) \\ V^{\pi_X}(\boldsymbol{X}_\ell) = \frac{1}{2}\boldsymbol{X}_\ell^T\, \boldsymbol{S}^X\, \boldsymbol{X}_\ell &\equiv J^{\pi_X}(\boldsymbol{X}_\ell), \end{aligned}$$

where $S^E \in \mathbb{R}^{o\times o}$ and $S^X \in \mathbb{R}^{n\times n}$ are solution matrices. They play a major role in the guided search policies determined using the online adaptive learning process.

The matrices S^E and S^X are computed using a temporal difference solution framework in real-time, and hence are relevant to the choice of the best policies π_E and π_X, respectively, using the interactive learning process.

The value functions V^{π_E} and V^{π_X} are utilized to form Bellman equations, which are temporal difference solution structures, for the underlying control mechanism, such that

$$\begin{aligned} V^{\pi_E}(\boldsymbol{E}_\ell) &= C^E(\boldsymbol{E}_\ell, \boldsymbol{u}_\ell^{\pi_E}) + V^{\pi_E}(\boldsymbol{E}_{\ell+1}) \\ V^{\pi_X}(\boldsymbol{X}_\ell) &= C^X(\boldsymbol{X}_\ell, \boldsymbol{u}_\ell^{\pi_X}) + V^{\pi_X}(\boldsymbol{X}_{\ell+1}). \end{aligned} \tag{4.2}$$

The Bellman equations encode the two interacting optimization processes that are solved simultaneously, where it is required to drive the tracking error $\boldsymbol{E}$ to zero and, at the same time, optimize the stability for the dynamics $\boldsymbol{X}$ along the navigation trajectories. Further, the Bellman equations (4.2) enable the integration between two mathematical

problems: The first is related to solving the navigation control problem, while the second enables an approximate dynamic programming solution (i.e., machine learning solution) for the problem. The online adaptive learning control process starts with an initial control strategy for each control decision and then the learning process, employing the above Bellman equations, directs the control strategies (i.e., learn better control decisions) using a value iteration process which is guaranteed to converge.

Herein, the error vector $\boldsymbol{E}_\ell$ is structured as follows:

$$\boldsymbol{E}_\ell = \begin{bmatrix} e_\ell & e_{\ell-1} & e_{v\ell} & e_{v\ell-1} & e_{s\ell} & s_{s\ell-1} \end{bmatrix}^T$$

where $e_{v\ell}$ and $e_{v\ell-1}$ are error derivatives, with respect to time, evaluated at time ℓ and $\ell - 1$, respectively. $e_{s\ell}$ and $s_{s\ell-1}$ are the moving averages calculated as

$$e_{s\ell} = \frac{1}{N} \sum_{i=\ell-N}^{\ell} e_i$$

and

$$e_{s\ell-1} = \frac{1}{N} \sum_{i=\ell-N-1}^{\ell-1} e_i,$$

respectively.

The way the error vector $\boldsymbol{E}_\ell$ is calculated, combines many sub-objectives which are optimized together. It minimizes the local tracking error e_ℓ while considering its previous instance $e_{\ell-1}$. Further, it modifies the rate of the control decision change by looking backward in-time to include error derivatives evaluated at time instances ℓ and $\ell - 1$, as well as evaluates average of the errors across longer time-intervals N. It is worth to mention that the vector formulation $\boldsymbol{E}_\ell$ enables more advanced forms of equivalent discrete Proportional-Integral-Derivative (PID) structures (i.e., dependence on $e_\ell, e_{\ell-1}$, and $e_{\ell-2}$) or even considers higher-order derivatives and integral equivalents (i.e., dependence on $e_\ell, e_{\ell-1}, e_{\ell-2}, \ldots$, and $e_{\ell-N}$).

The solution of the optimal tracking problem relies on solving a number of differential equations, a subgroup of which is solved offline and at the same time they do not allow dynamical forms of the error signals. Herein, the formulation of the adaptive learning mechanism allows for an online solution as well as using a variety of tracking error dynamical forms. The experimental setup, described in Section 4.3, provides measurements

related to the orientation of the wing. This made it convenient to choose the states $\boldsymbol{X}_\ell$ as

$$\boldsymbol{X}_\ell = \begin{bmatrix} \theta_\ell & \theta_{v\ell} & \theta_{a\ell} \end{bmatrix}^T,$$

where $\theta_\ell, \theta_{v\ell}$, and $\theta_{a\ell}$ are the pitch attitude, pitch velocity, and pitch acceleration, respectively.

The desired control policies $\boldsymbol{\pi}_E$ and $\boldsymbol{\pi}_X$ are linear feedback policies which are used to decide on the different control signals, such that

$$\boldsymbol{u}_\ell^{\pi_E} = \boldsymbol{\pi}_E \, \boldsymbol{E}_\ell, \quad \boldsymbol{u}_\ell^{\pi_X} = \boldsymbol{\pi}_X \, \boldsymbol{X}_\ell. \tag{4.3}$$

The next section will explain how to compute the real-time negative feedback control strategies $\boldsymbol{\pi}_E$ and $\boldsymbol{\pi}_X$.

4.5 Machine learning platform

The online machine learning framework is driven by the design of the control problem. First, the computational platform based on a value iteration implementation for an online reinforcement learning solution is detailed. Then, adaptive critics are employed to provide neural network implementation for the approximate dynamic programming solution.

4.5.1 Guided search process

The policies $\boldsymbol{\pi}_E$ and $\boldsymbol{\pi}_X$ are guided toward the intended optimization objectives stated by the designer as follows:

$$\boldsymbol{u}^{\pi_E} = -\boldsymbol{P}^E \frac{\partial V^{\pi_E}(\boldsymbol{E}_\ell)}{\partial \boldsymbol{E}_\ell}, \quad \boldsymbol{u}^{\pi_X} = -\boldsymbol{P}^X \frac{\partial V^{\pi_X}(\boldsymbol{X}_\ell)}{\partial \boldsymbol{X}_\ell}, \tag{4.4}$$

where $\boldsymbol{P}^E \in \mathbb{R}^{m \times o}$ and $\boldsymbol{P}^X \in \mathbb{R}^{s \times n}$ are guiding search vectors which carry the intentions and dynamic preferences of the designer.

The vectors $\boldsymbol{P}^E$ and $\boldsymbol{P}^X$ contain selective dynamic forms (i.e., introduced by the designer to reflect their objectives regarding the entries in vectors $\boldsymbol{E}$ and $\boldsymbol{X}$) as will be highlighted

in the experimental analysis. This in turn promotes flexibility for a new class of online learning processes with guided search features.

The control policies $\boldsymbol{\pi}_E = -\boldsymbol{P}^E\,\boldsymbol{S}^E$ and $\boldsymbol{\pi}_X = -\boldsymbol{P}^X\,\boldsymbol{S}^X$, i.e., these associated with

$$\frac{\partial V^{\pi_E}(\boldsymbol{E}_\ell)}{\partial \boldsymbol{E}_\ell}, \frac{\partial V^{\pi_X}(\boldsymbol{X}_\ell)}{\partial \boldsymbol{X}_\ell},$$

are applied in real-time using a value iteration process as will be explained next.

4.5.2 Value iteration algorithm

An online reinforcement learning solution is developed using the aforementioned Bellman equations (4.2). The solution is realized using a two-step value iteration process. The first updates the solving value functions (i.e., $\boldsymbol{S}^E$ and $\boldsymbol{S}^X$) using (4.2) while the second extracts the new or improved policies (i.e., $\boldsymbol{\pi}_E$ and $\boldsymbol{\pi}_X$). The online value iteration process is detailed in Algorithm 1. Bellman equations provide the temporal difference form necessary to adapt according to the online learning process beyond the initially selected policies $\boldsymbol{\pi}_E^0$ and $\boldsymbol{\pi}_X^0$. Note that, unlike policy iteration paradigms, these initial policies do not need to be admissible. Hence, the intentionally guided policies are improved along the trajectory of the system.

Value iteration processes are proven to converge in general for single and multi-agent systems based on Lyapunov stability approaches if the underlying systems are stabilizable [98, 130, 133]. This is mainly due to the properties of convex objective cost functions (i.e., C^E and C^X in this work) and consequently Bellman solution forms (i.e., (4.2)). Hence, the value functions evolve in a bounded manner, such that

$$\begin{aligned} 0 \leq V^{\pi_E(0)}(\boldsymbol{E}_\ell) \leq V^{\pi_E(1)}(\boldsymbol{E}_\ell) \leq \cdots \leq V^{\pi_E(t)}(\boldsymbol{E}_\ell) \leq \cdots \leq V^{\pi_E(*)}(\boldsymbol{E}_\ell), \\ 0 \leq V^{\pi_X(0)}(\boldsymbol{X}_\ell) \leq V^{\pi_X(1)}(\boldsymbol{X}_\ell) \leq \cdots \leq V^{\pi_X(t)}(\boldsymbol{X}_\ell) \leq \cdots \leq V^{\pi_X(*)}(\boldsymbol{X}_\ell), \end{aligned} \tag{4.5}$$

where the value functions $V^{\pi_E(*)}$ and $V^{\pi_X(*)}$ are the optimal response solutions for Bellman equations (4.2). In other words, the tracking error and motion dynamics are stable if the aircraft system is controllable. The online value iteration learning process guides the solving value functions toward the best values $V^{\pi_E(*)}(\boldsymbol{E}_\ell)$ and $V^{\pi_X(*)}(\boldsymbol{X}_\ell)$, and hence best guided policies $\boldsymbol{\pi}_E^*$ and $\boldsymbol{\pi}_X^*$.

4.5.3 Adaptive critics implementation

Adaptive critics are employed as neural network approximation methods for the presented online reinforcement learning solution [100]. The adaptive critics scheme is implemented using means of neural network structures, namely actor-critic networks, for each optimization control process. They provide solutions for the underlying Bellman equations using the guided search mechanism. The critic and actor structures approximate the solving value functions and the associated policies in an interactive manner implicitly during iterative updates of the underlying Bellman equations. The actor network reflects the improvements in the guided control strategy while its quality is approximated by the critic network. The weights of the actor-critic structures are adapted using a gradient descent approach motivated by the linear forms of the control policies similar to those that appear in [22].

The value functions V^{π_E} and V^{π_X} (i.e., the critic structures) are approximated so that

$$\hat{V}^{\pi_E}(\boldsymbol{E}_\ell) = \frac{1}{2}\boldsymbol{E}_\ell^T\,\boldsymbol{\Omega}_c^E\,\boldsymbol{E}_\ell, \text{and} \hat{V}^{\pi_X}(\boldsymbol{X}_\ell) = \frac{1}{2}\boldsymbol{X}_\ell^T\,\boldsymbol{\Omega}_c^X\,\boldsymbol{X}_\ell,$$

where $\boldsymbol{\Omega}_c^E \in \mathbb{R}^{o\times o}$ and $\boldsymbol{\Omega}_c^X \in \mathbb{R}^{n\times n}$ are the critic approximation weights of the value functions $\hat{V}^{\pi_E}(\boldsymbol{E}_\ell)$ and $\hat{V}^{\pi_X}(\boldsymbol{X}_\ell)$, respectively. The critic network forms are motivated by the structures of the value functions V^{π_E} and V^{π_X} respectively.

In a similar fashion, the guided search policies (i.e., the actor structures) are approximated so that

$$\hat{\boldsymbol{u}}_\ell^{\pi_E} = \boldsymbol{\Omega}_a^E\boldsymbol{E}_{\boldsymbol{\ell}}, \quad \hat{\boldsymbol{u}}_\ell^{\pi_X} = \boldsymbol{\Omega}_a^X\boldsymbol{X}_{\boldsymbol{\ell}},$$

where $\boldsymbol{\Omega}_a^E \in \mathbb{R}^{m\times o}$ and $\boldsymbol{\Omega}_a^X \in \mathbb{R}^{s\times n}$ are the actor approximation weights of the policies $\boldsymbol{\pi}_E$ and $\boldsymbol{\pi}_X$, respectively.

The different approximation weights are updated using a gradient descent approach that applies a minimization criteria on the squared approximation errors. The approximation errors of the critic networks have squared forms, so that

$$\begin{aligned}\varepsilon_c^{E_\ell} &= \frac{1}{2}\left(\hat{V}^{\pi_E}(\boldsymbol{E}_\ell) - \hat{V}^{T_E}(\boldsymbol{E}_\ell)\right)^2,\\ \varepsilon_c^{X_\ell} &= \frac{1}{2}\left(\hat{V}^{\pi_X}(\boldsymbol{X}_\ell) - \hat{V}^{T_X}(\boldsymbol{X}_\ell)\right)^2,\end{aligned} \tag{4.6}$$

where the target values $\hat{V}^{T_E}(\boldsymbol{E}_\ell)$ and $\hat{V}^{T_X}(\boldsymbol{X}_\ell)$ are calculated using

$$\hat{V}^{T_E}(\boldsymbol{E}_\ell) = C^E(\boldsymbol{E}_\ell, \hat{\boldsymbol{u}}_\ell^{\pi_E}) + \hat{V}^{\pi_E}(\boldsymbol{E}_{\ell+1})$$
$$\hat{V}^{T_X}(\boldsymbol{X}_\ell) = C^X(\boldsymbol{X}_\ell, \hat{\boldsymbol{u}}_\ell^{\pi_X}) + \hat{V}^{\pi_X}(\boldsymbol{X}_{\ell+1}).$$

The critic weights are adapted according to a gradient descent rule such that

$$\begin{aligned}\boldsymbol{\Omega}_c^{E(t+1)} &= \boldsymbol{\Omega}_c^{E(t)} - \alpha_c^E \left[\frac{\partial \varepsilon_c^{E_\ell}}{\partial \boldsymbol{\Omega}_c^E}\right]^{(t)} \\ \boldsymbol{\Omega}_c^{X(t+1)} &= \boldsymbol{\Omega}_c^{X(t)} - \alpha_c^X \left[\frac{\partial \varepsilon_c^{X_\ell}}{\partial \boldsymbol{\Omega}_c^X}\right]^{(t)},\end{aligned} \tag{4.7}$$

where $0 < \alpha_c^E, \alpha_c^X < 1$ are critic networks learning rates,

$$\begin{aligned}\frac{\partial \varepsilon_c^{E_\ell}}{\partial \boldsymbol{\Omega}_c^E} &= \left(\hat{V}^{\pi_E}(\boldsymbol{E}_\ell) - \hat{V}^{T_E}(\boldsymbol{E}_\ell)\right) \boldsymbol{E}_\ell \boldsymbol{E}_\ell^T, \\ \frac{\partial \varepsilon_c^{X_\ell}}{\partial \boldsymbol{\Omega}_c^X} &= \left(\hat{V}^{\pi_X}(\boldsymbol{X}_\ell) - \hat{V}^{T_X}(\boldsymbol{X}_\ell)\right) \boldsymbol{X}_\ell \boldsymbol{X}_\ell^T,\end{aligned}$$

and t refers to the iteration update index.

Similarly, the squared approximation errors for the actor networks are defined by

$$\varepsilon_a^{E_\ell} = \frac{1}{2}\left(\hat{\boldsymbol{u}}_\ell^{\pi_E} - \hat{\boldsymbol{u}}_\ell^{T_E}\right)^2, \quad \varepsilon_a^{X_\ell} = \frac{1}{2}\left(\hat{\boldsymbol{u}}_\ell^{\pi_X} - \hat{\boldsymbol{u}}_\ell^{T_X}\right)^2,$$

where the target values $\hat{\boldsymbol{u}}_\ell^{T_E}$ and $\hat{\boldsymbol{u}}_\ell^{T_X}$ are calculated as follows

$$\hat{\boldsymbol{u}}_\ell^{T_E} = -\boldsymbol{P}^E \boldsymbol{\Omega}_c^E \boldsymbol{E}_\ell, \quad \hat{\boldsymbol{u}}_\ell^{T_X} = -\boldsymbol{P}^X \boldsymbol{\Omega}_c^X \boldsymbol{X}_\ell.$$

Using gradient descent, the actor network weights are adapted according to the following rule

$$\begin{aligned}\boldsymbol{\Omega}_a^{E(t+1)} &= \boldsymbol{\Omega}_a^{E(t)} - \alpha_a^E \left[\frac{\partial \varepsilon_a^E}{\partial \boldsymbol{\Omega}_a^{E_\ell}}\right]^{(t)} \\ \boldsymbol{\Omega}_a^{X(t+1)} &= \boldsymbol{\Omega}_a^{X(t)} - \alpha_a^X \left[\frac{\partial \varepsilon_a^X}{\partial \boldsymbol{\Omega}_a^{X_\ell}}\right]^{(t)},\end{aligned} \tag{4.8}$$

where $0 < \alpha_a^E, \alpha_a^X < 1$ are actor networks learning rates, and

$$\frac{\partial \varepsilon_a^{E_\ell}}{\partial \boldsymbol{\Omega}_a^E} = \left(\hat{\boldsymbol{u}}_\ell^{\pi_E} - \hat{\boldsymbol{u}}_\ell^{T_E}\right) \boldsymbol{E}_\ell^T,$$
$$\frac{\partial \varepsilon_a^{X_\ell}}{\partial \boldsymbol{\Omega}_a^X} = \left(\hat{\boldsymbol{u}}_\ell^{\pi_X} - \hat{\boldsymbol{u}}_\ell^{T_X}\right) \boldsymbol{X}_\ell^T.$$

The full schematic diagram of the adaptive critics process is shown in Figure 4.4 for the combined control problem in hand. The different actor and critic weights are updated simultaneously in real-time following Algorithm 1.

Algorithm 1: Online Value Iteration Process.

Input: Desired trajectory $\theta_\ell^{desired}$, weighting matrices $\boldsymbol{Q}^E, \boldsymbol{R}^E, \boldsymbol{Q}^X$, and $\boldsymbol{R}^X$, guiding search vectors $\boldsymbol{P}^E$ and $\boldsymbol{P}^X$, tracking error evaluation interval N.

Output: Policies $\boldsymbol{\pi}_E$, $\boldsymbol{\pi}_X$, and tracking error e_ℓ, for $\ell = 0, 1, \ldots$

begin

  $\ell = 0, t = 0$ `/* time and strategy indices */`

  Initialize solving matrices $\boldsymbol{S}^{E\{0\}}$ and $\boldsymbol{S}^{X\{0\}}$ `/* Positive definite */` and hence initial policies $\boldsymbol{\pi}_E^0$ and $\boldsymbol{\pi}_X^0$ and tracking errors interval $e_{\ell-1}, \ldots, e_{\ell-N-1}$

  Measure $\theta_\ell, \theta_{v\ell}, \theta_{a\ell}$ and then calculate the errors e_ℓ, $e_{\ell-1}$, $e_{v\ell}$, $e_{v\ell-1}$, $e_{s\ell}$, $e_{s\ell-1}$

  repeat`/* Training/Search loop */`

    Compute the different control signals $u^{\pi_E(t)}$ and $u^{\pi_X(t)}$ using (4.4)

    Measure $\theta_{\ell+1}, \theta_{v\ell+1}, \theta_{a\ell+1}$ and then calculate the errors $e_{\ell+1}$, e_ℓ, $e_{v\ell+1}$, $e_{v\ell}$, $e_{s\ell+1}$, $e_{s\ell}$

    Evaluate the solving value functions
    $V^{\pi_E(t+1)}(\boldsymbol{E}_\ell) = C^E(\boldsymbol{E}_\ell, \boldsymbol{u}_\ell^{\pi_E(t)}) + V^{\pi_E(t)}(\boldsymbol{E}_{\ell+1})$
    $V^{\pi_X(t+1)}(\boldsymbol{X}_\ell) = C^X(\boldsymbol{X}_\ell, \boldsymbol{u}_\ell^{\pi_X(t)}) + V^{\pi_X(t)}(\boldsymbol{X}_{\ell+1}).$

    Extract the improved control policies
    $\boldsymbol{\pi}_E{}^{t+1} = -\boldsymbol{P}^E S^{E(t+1)}, \boldsymbol{\pi}_X{}^{t+1} = -\boldsymbol{P}^X S^{X(t+1)},$

    $\ell \leftarrow \ell + 1$ `/* Update real-time index */`

    $t \leftarrow t + 1$`/* Update policy index */`

  until *Satisfactory trajectory-tracking performance (i.e., acceptable tracking error).*

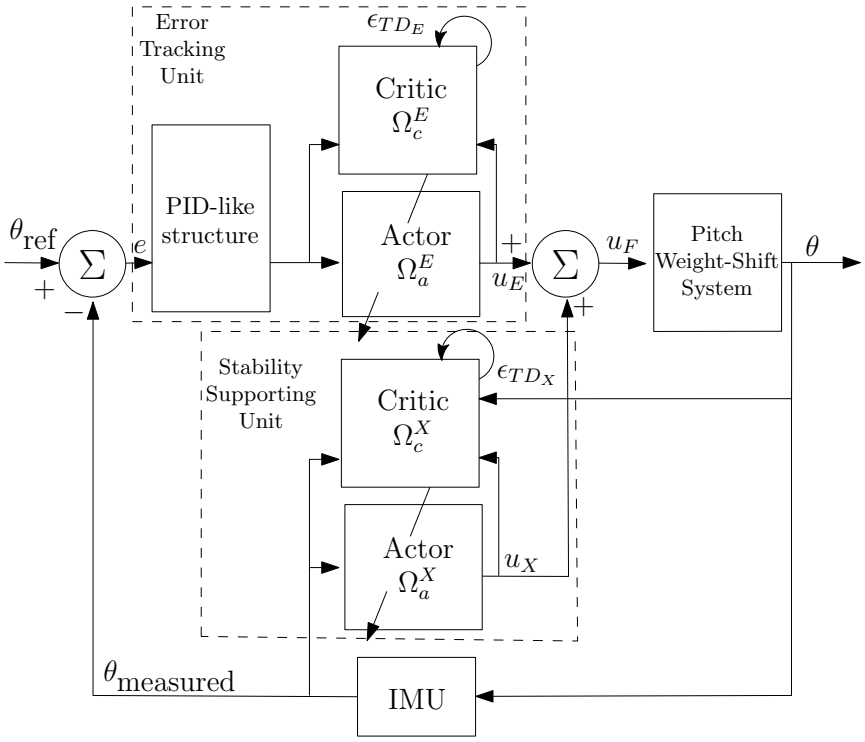


Figure 4.4: Feedback control loop of pitch weight-shift control mechanism. The adaptive error tracking unit and stability support unit cooperate to achieve the trajectory tracking objective.

4.5.4 Scalability of the learning mechanism

Algorithm 1 possesses several merits for the tracking problems. First, the learning process can be conditioned for any number of tracking signals simultaneously. As an example, a large tracking or navigation process that contains many tracking objectives would be easy to implement compared to solving a high number of differential equations in an offline mode. Second, the tracking error signals could involve many dynamical forms (i.e., moving velocity, acceleration, moving average, etc) which were not possible or easy-to-implement before. Third, this scheme allows for coupled optimized dynamical processes, such as optimization of multiple coupled and interactive dynamical objectives for multiple problems without the need to independently solve each problem. Finally, the arrangement of Bellman equations enables simple and straightforward adaptive critics solution, since the control strategies hold linear forms.

4.6 Experimental results

The online model-free adaptive learning mechanism is validated in a real-time environment using a two parallel servomotor pitch actuation system. The system generates servomotor pull actions dependent on the current-measured pitch position and the desired trajectory steering the wing. For the sake of experiments presented below, the trajectory reference is chosen to be sinusoidal to provide continuous pitch up/down commands.

4.6.1 Learning parameters

The learning parameters of the system are selected in order to reflect the physical constraints of the measured dynamical parameters and also encode the control design preferences for the desired system response. They are chosen as

$$
\begin{aligned}
R^E &= 10^{-7}, \\
R^X &= 10^{-7}, \\
Q^E &= 10^{-4} \begin{bmatrix} 25 & 0 & 0 & 0 & 0 & 0 \\ 0 & 25 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.25 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.25 & 0 & 0 \\ 0 & 0 & 0 & 0 & 25 & 0 \\ 0 & 0 & 0 & 0 & 0 & 25 \end{bmatrix}, \\
Q^X &= 10^{-6} \begin{bmatrix} 25 & 0 & 0 \\ 0 & 25 & 0 \\ 0 & 0 & 0.0025 \end{bmatrix}.
\end{aligned}
$$

The weighting matrices Q^E, Q^X, R^E, and R^X are selected to balance the physical constraints of the states θ_ℓ, $\theta_{v\ell}$, $\theta_{a\ell}$, e_ℓ and the control input u^F with the designer's preferences for achieving the desired system response. Specifically, the diagonal elements of Q^E and Q^X are assigned higher values (e.g., 25) for states that require tighter regulation to ensure stability and performance, while lower values (e.g., 0.25 and 0.0025) are used for less critical states that have potential for a larger range of values. The control weighting matrices R^E and R^X are set to 10^{-7} to minimize the penalization on control effort, allowing the controller to make necessary adjustments without being overly restricted by the cost on the control signals.

Vectors P^E and P^X for the guided search policies π^E and π^X are selected so that $P^E = \begin{bmatrix}200 & 50 & 10 & 5 & 10 & 5\end{bmatrix}$ and $P^X = \begin{bmatrix}10 & 10 & 5\end{bmatrix}$. The vector P^E assigns more weight to minimizing the recently measured tracking errors e_ℓ and $e_{\ell-1}$, as opposed to the error velocity terms $e_{v\ell}$ and $e_{v\ell-1}$, to lessen the abruptness of potential changes stemming from nonlinear transitions. Furthermore, the influences of the moving average tracking errors $e_{s\ell}$ and $e_{s\ell-1}$ on the overall performance are weighted similarly to those of the error velocities. The vector P^X reflects the gradual dynamic importance of the pitch attitude θ_ℓ and the pitch velocity $\theta_{v\ell}$ over the angular acceleration $\theta_{a\ell}$. The actor and critic learning rates are selected to be small enough in order to match the actor and critic adjustments in a smooth manner. They are set to $\alpha_c^E = 0.01$, $\alpha_c^X = 0.01$, $\alpha_a^E = 0.01$, and $\alpha_a^X = 0.05$. The higher value for α_a^X is chosen to reduce the sensitivity of the stability supporting agent control policy to sudden changes in the value function approximation. The desired navigation trajectory is an independent sinusoidal reference signal that takes the form $\theta_\ell^{desired} = 20\,\sin(0.132\,\pi\,\ell)$.

4.6.2 Test scenarios

The adaptive learning controller is validated using three test scenarios. In the first scenario, the system is tested under nominal circumstances where no external disturbance is applied on the system. The dynamic performance of the proposed adaptive learning controller is shown in Figure 4.5. This experiment reveals that the learning algorithm successfully converges to a stable control policy which prescribes correct servomotor pull-forces for the observed system state, i.e., pitch attitude. The absolute average tracking error over the test period is found to be $0.45\,^\circ$. It is worth noticing that the convergence behaviour of the proposed control mechanism is not affected by the occasional sudden erroneous readings in the feedback measurements, as revealed at around time instants 80 s and 123 s in Figure 4.5(a). Such errors are due to measurement noise in the sensing units that are not attenuated by the low-pass filter. The convergence of the adaptive learning approach is clear in Figure 4.6, where the actor and critic weights converge throughout the online learning process. The overall normalized forces generated by the servomotors are shown in Figure 4.5(b). Finally, the convergence of the value iteration approach, as described in Section 4.5, is highlighted in Figure 4.7. It is shown that a general bounded non-decreasing convergence pattern is observed, despite the sensor reading disturbances, as expected for the underlying value iteration process.

The second scenario is set up to highlight the ability of the controller to reject sudden external and forced disturbances on the system, through demonstration. The test is a repetition of the first, except that for this case an intentional disturbance was applied at around 120 s for about 10 s by manually agitating the control bar around the pitch and roll axes to stimulate the effects of turbulent wind gusts or crazed pilot input on the wing during the flight. The tracking performance of the proposed controller is demonstrated in Figure 4.8(a). The controller is started with converged control gains from a previous learning episode. As such, minimal changes in the critic weights are witnessed, as shown by the critic weights evolution in Figure 4.9. During disturbance, the critic reacts appropriately to generate large abrupt motor torques to counteract the perturbations induced by the manual disturbance, as shown in Figure 4.8(b). Once the dynamics are back to their nominal conditions when the disturbance subsided, the controller successfully resumes the tracking of the commanded pitch. After the disturbance, the critic adjusts its weights to incorporate new information learned from the disturbance's interaction with the system, but these adjustments are not reflective of nominal operation, so the actor converges back to the policy observed before the disturbance.

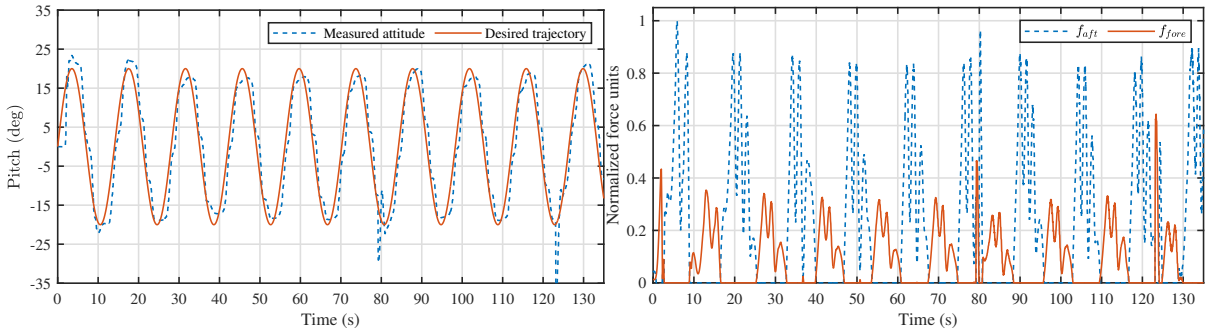
The goal of the third experiment is to show that once the controllers converge to some optimal strategies π_E^* and π_X^*, the actor's learning mechanism can be turned off without degrading the control performance. By doing so, the controller is made to operate with static control gains. For this test case, the adopted static control policies applied are those converged from a prior experiment. The controller's performance is illustrated in Figure 4.10. The absolute average tracking error over the test period is found to be 0.94 °. This result emphasizes that the converged strategies from static "actor-only" controllers previously derived from the adaptive learning mechanism remain valid during the trajectory tracking process. Notice how this experiment also witnessed two abrupt measurement disturbances in the sensor feedback.

Unlike classical Q-leaning processes which employ multiple offline training episodes before settling on suitable control strategies, herein the proposed approach showed successful outcomes following a single real-time learning episode as shown in Figures 4.5(a), 4.8(a). After 30 s of real-time online reinforcement learning elapsed, the system converges towards optimal trajectory tracking policies. A clear difference is observed when comparing control input signals of the fore and aft parallel actuators, a result of the optimal control policy. Unintended physical impulse disturbances at 80 s and 120 s are captured by the critics with step changes in their weights, but these negligibly affect the actor policies. When the converged to policies are frozen and then used to run the system as shown in Figure 4.10(a), the proposed approach is shown to follow desired trajectories with bounded errors. The

maximum observed absolute average tracking error was kept to under 1° for the mechanical actuation system, without overlooking the sensory error bias.

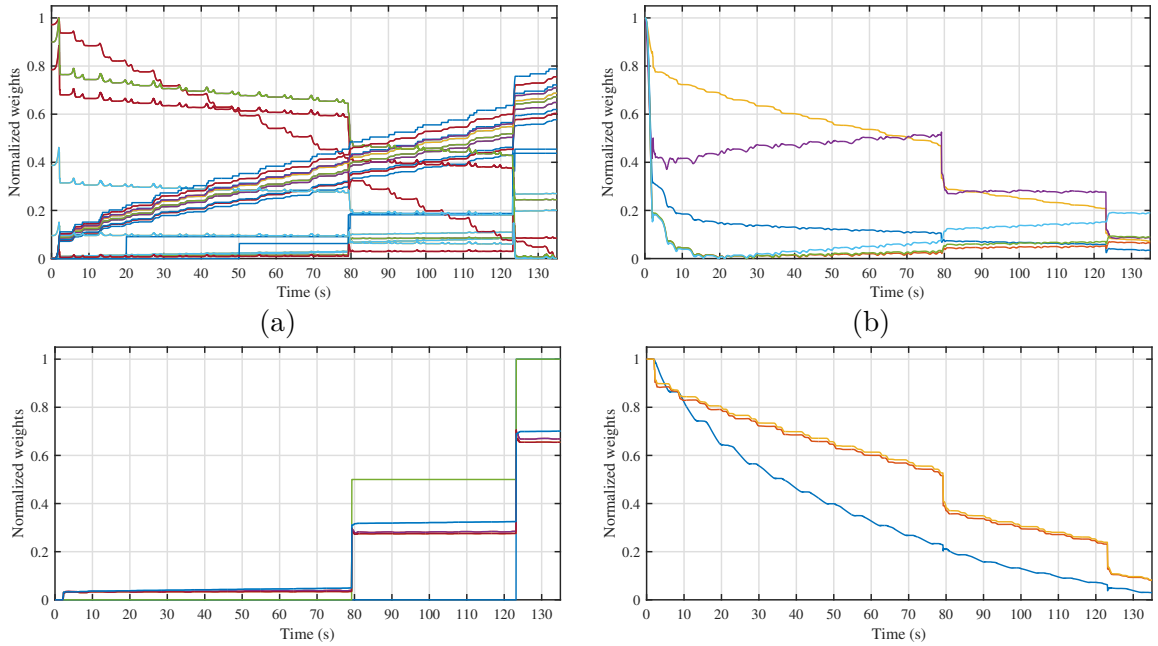
When experimenting with real systems, as opposed to simulation, it is typical to notice differences between the reference and the actual signals. The differences may be due to several reasons, including noise, delay caused by digital filters, and unmodelled dynamic nonlinearities like gear backlashes. In this particular work, the differences also come from the coupling between the roll and pitch motions. The two were assumed to be decoupled in the first-order linear approximation of the system in [19]. However, in a real weight-shift system, they cannot be completely decoupled. Nevertheless, it is clear from the figures that the lag between the reference and the actual signals is bounded within an acceptable margin.

Figures 4.5(a), 4.8(a), and 4.10(a) emphasize the bounded stability features and the ability of the online learning mechanism to maintain stability under significant mechanical disturbances, noise, and measurement errors. These results show that the tracking errors are bounded and non-increasing. This point is further supported by examining the resulting actuation input signals shown in Figures 4.5(b), 4.8(b), and 4.10(b).



(a) (b)

Figure 4.5: Control performance during online learning process: (a) measured vs. desired pitch attitude, (b) acting forces on the wing's keel, f_{fore} and f_{aft}.



(a) (b) (c) (d)

Figure 4.6: Evolution of actor-critic weights during learning process: (a) tracking critic unit, (b) tracking actor unit, (c) stabilizing critic unit, (d) stabilizing actor unit.

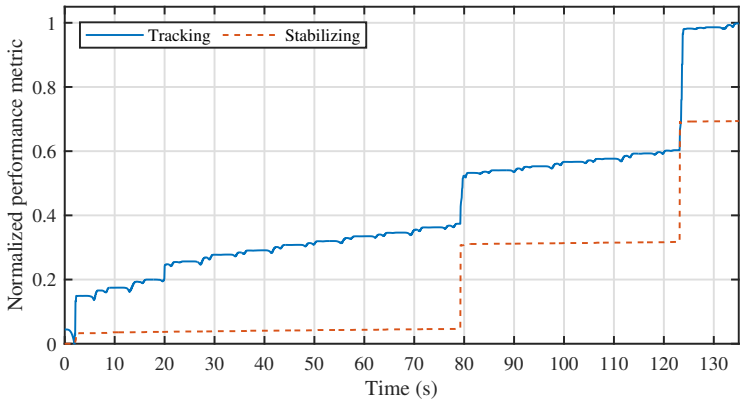
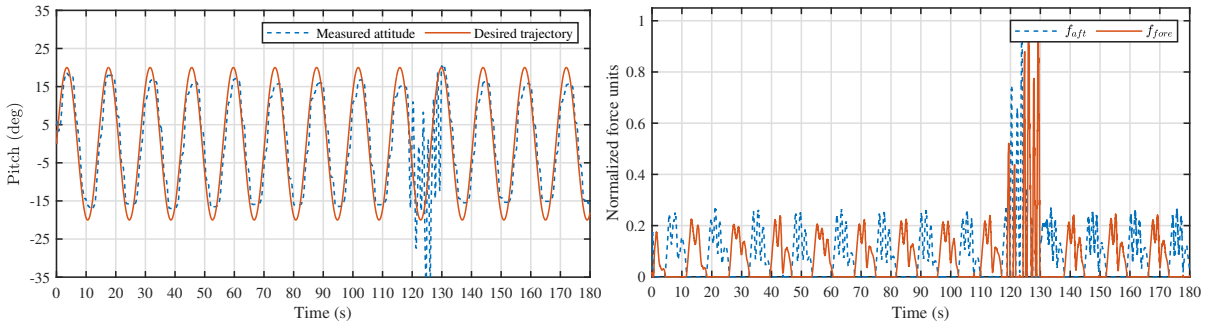
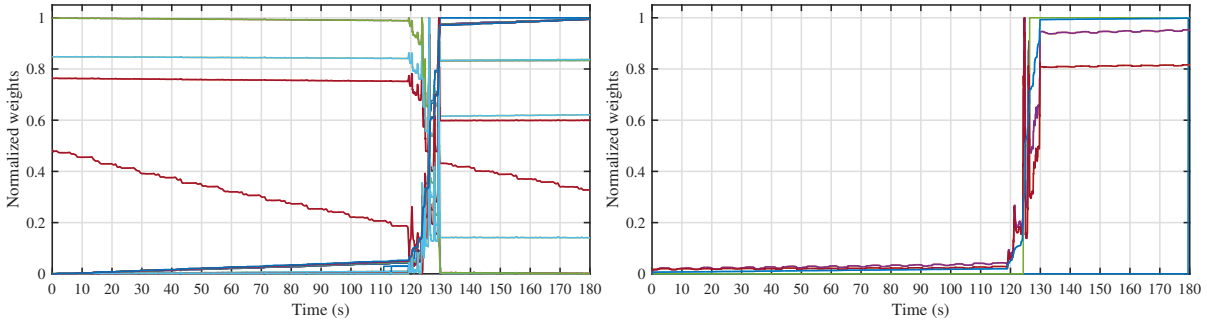


Figure 4.7: Evolution of the policy evaluation metric.



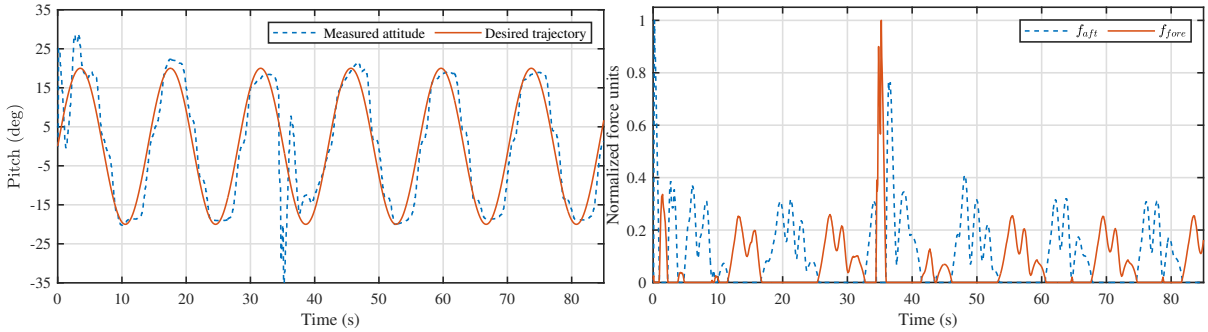
(a) (b)

Figure 4.8: Control performance while learning with a converged controller in the presence of mechanical disturbances: (a) measured vs. desired pitch attitude, (b) acting forces on the wing's keel, f_{fore} and f_{aft}.



(a) (b)

Figure 4.9: Variations in critic weights in the face of mechanical disturbances: (a) tracking unit, (b) stabilizing unit.



(a) (b)

Figure 4.10: Control performance with previously learned static control policies: (a) measured vs. desired pitch attitude, (b) acting forces on the wing's keel, f_{fore} and f_{aft}.

4.7 Chapter summary

This chapter introduced a solution to trajectory tracking control for a weight-shift kinematic mechanism surrogate by integrating a model-free reinforcement learning controller directly with a set of flight controller hardware, instrumentation, and parallel servo-winch actuators. The feedback loop receives the inertial measurements from the surrogate system, and learns in real-time the optimal control strategies without any prior knowledge of the system's dynamics. Issues could arise with a model-based strategy, as the aircraft's nonlinear dynamics inherently lead to sensitivity to parameter uncertainty, potentially resulting in poor control without online adaptation. In fact, the work demonstrated that for instances when the surrogate mechanism is subjected to severe disturbances, the controller continues to perform without destabilizing the system. The quality of the trajectory tracking controller depends on the design of the model-free learning process rather than the precision or number of sensors providing feedback on the total system state. The online learning solution for optimal trajectory tracking, along with the technique of guided search, and their deployment to parallel servo-winch actuators provide a foundation for furthering development of a full weight-shift flight control piloting system.

Chapter 5

uWSC Aircraft Simulator: A Gazebo-Based Model For Uncrewed Weight-Shift Control Aircraft Flight Simulation [3]

5.1 Foreword

This chapter presents a robot simulation methodology to model the flight dynamics of a weight-shift aircraft using open-source software. It is adapted from the paper titled "uWSC Aircraft Simulator: A Gazebo-Based Model for Uncrewed Weight-Shift Control Aircraft Flight Simulation" that was published in the proceedings of the 2023 IEEE International Symposium of Robotic and Sensors Environments (ROSE), held on November 6-7th of 2023, in Tokyo, Japan. A copy of the original paper is included in Appendix A.3.

The paper's primary contribution is an open-source robotics model of the industry partner's prototype unmanned aircraft system, referred to as the uncrewed weight-shift control (uWSC) model. To prove its viability as an appropriate development tool, the performance of the uWSC aircraft model was benchmarked against data captured from real weight-shift aircraft flight tests. The results indicated that the model provides a suitable approximation of real weight-shift aircraft flight. This tool has been used internally by the industry

partner to bridge the gap between high-level flight controller development and low-level weight-shift control dynamics.

The work was motivated by the need for a weight-shift flight controller development environment, to pre-validate their performance prior to real test flights of the weight-shift aircraft. By employing a simulation methodology, the development and study of new flight controls could be studied without performing real flight tests. At the inception of this work, Romaeris had performed preliminary studies of dynamics modelling of the weight-shift aircraft internally, based on previous approaches found in the literature [19, 20, 152]. It was concluded that it was difficult to adequately fit these models to real WSC aircraft test data due to the large set of parameters required to describe the nonlinear dynamics of the aircraft. This leads to a circular problem, where without test flights to collect data, flight control developments had slowed or stalled, and the flight test of controls could only be performed in manned test with strictly limited control authority, limiting the amount of data that could be captured. The simulation project was predicted to be a worthwhile research direction that would deliver immediate benefits to the flight controller engineering if it could be realized.

By using Gazebo, the essential dynamics of the weight-shift aircraft prototype can be constructed through the integration of physics-based modelling components. However, when it comes to the flexible wing's aerodynamics, there is notable limitations on the modelling fidelity. It would be extremely difficult to develop a true aeroelastic model within Gazebo, without substantial development. As a compromise, the lateral and longitudinal aerodynamic stability properties expected for hang-glider wings are realized by joining multiple rigid wing sections. This idea finds its roots in the modelling of flexible wings as sets of rigid wing sections, as developed previously for hang-glider wings in [19], and in Gazebo simulation in [227]. It was found by comparative analysis through the replay of weight-shift flight missions in simulation that the multi-section rigid wing model approximated the flight trajectories of the real aircraft reasonably well.

New prototype aircraft designs developed by Romaeris integrate robotic piloting systems to enable uWSC flight, unlocking their use as remotely piloted aircraft systems. Figure 5.1 presents a capture from simulation of the model performing a simulated flight. This model has been contributed to the official ArduPilot open-source simulator models repository with the aim that it serves as a critical tool to support further study of uWSC aircraft control systems [40].

Figure 5.1: Capture from Gazebo simulator of the uWSC aircraft model performing a starboard side banking turn. The hung mass under the wing is responsible for attitude adjustments via weight-shifting control.

The chapter is organized as follows: The dynamics model for the uWSC aircraft simulator model based on the prototype design is presented in Section 5.2. In Section 5.3 a set of benchmarks is defined to quantify flight performance of the simulated WSC flight against a set of data collected from real WSC aircraft flight test.

5.2 Weight-shift aircraft Gazebo model

The uWSC aircraft simulator model developed using Gazebo is based on the weight-shift aircraft prototype in Figure 5.2. The aircraft prototype consists of two primary bodies: the wing structure which includes the landing gear, and the weight-shifting mass that hangs beneath the wing which includes the batteries, propellers and payload. The two bodies are attached by a two-axis revolute joint on the wing keel named the hang-block, allowing the shifting of hung mass to produce attitude control by weight-shift adjustments. The software-in-the-loop feature of ArduPlane is employed to simulate automatic flight

Figure 5.2: 3D CAD model of the Romaeris uncrewed weight-shift aircraft prototype.

control [229, 234]. Attitude adjustment of the wing via fuselage weight-shifting is induced by direct control of the hang-block joint angles.

The Gazebo model approximates the prototype aircraft's mechanical subsystems. Table 5.3 presents the Gazebo model's primary components' mass, inertia, and relative center of mass properties that were calculated from a 3D computer-aided design model. The layout of the primary components is visualized with RVIZ in Figure 5.3, and the corresponding tree diagram visualization of the model's link structure is presented by Figure 5.4 [235].

The prototype aircraft is equipped with a Wills Wing Falcon 3 170 [239] that is rated for a maximum velocity of 77 km/h and a maximum takeoff load of 130 kg. Expert weight-shift pilot [28] and aerodynamic modelling references are consulted to derive a reasonable approach to approximate the wing's aerodynamic stability properties as a rigid component within Gazebo [8, 19, 152, 227]. The approach taken to model the correct longitudinal dynamic stability features of the flexible wing is summarized as follows:

Table 5.1: The Unified Robot Description Format (URDF) body structure name field entries.

Acronym	**Meaning**
IMU	Inertial Measurement Unit: A sensor module that measures linear acceleration and angular velocity, and sometimes orientation, to help track a robot's motion.
XYZ	A standard Cartesian coordinate system used to define positions in 3D space.
RPY	Roll-Pitch-Yaw: A rotation parameterization defining the orientation of a robot or component in 3D using three angles.

Table 5.3: Mass, inertia tensor, and relative center of mass location for primary components of the uWSC aircraft Gazebo model

Link name	**Mass** $[kg]$	**Inertia** $[kg \cdot m^2]$						**Relative position** $[m]$		
		Ixx	Ixy	Ixz	Iyy	Iyz	Izz	X	Y	Z
Wing root, left	8.0	102.6	0.09	0	9.56	-3.1e-3	112.14	-0.5	0	-0.75
Wing root, right	8.0	102.6	0.09	0	9.56	-3.1e-3	112.14	0.5	0	-0.75
Wing tip, left	3.0	41.1	0.03	0	3.82	-1.2e-3	44.96	-0.5	0	0
Wing tip, right	3.0	41.1	0.03	0	3.82	-1.2e-3	44.96	0.5	0	0
Landing gear	18.8	2.05	0.03	-0.08	4.36	-1.6e-2	3.80	0.5	-0.03	-1.2
Fuselage	14.6	0.56	0	-0.16	2.68	0	2.67	0	0	-1
Battery	17.1	0.32	0	0	0.6	0	0.35	0	0	-0.5
Motor	1.74	0.01	0	0	3.0e-3	0	3.0e-3	0	0	0
Hang-block	0.64	0	0	0	0	0	0	0	0	0
Propeller	0.23	0	0	0	1.68e-2	0	1.68e-2	0	0	0

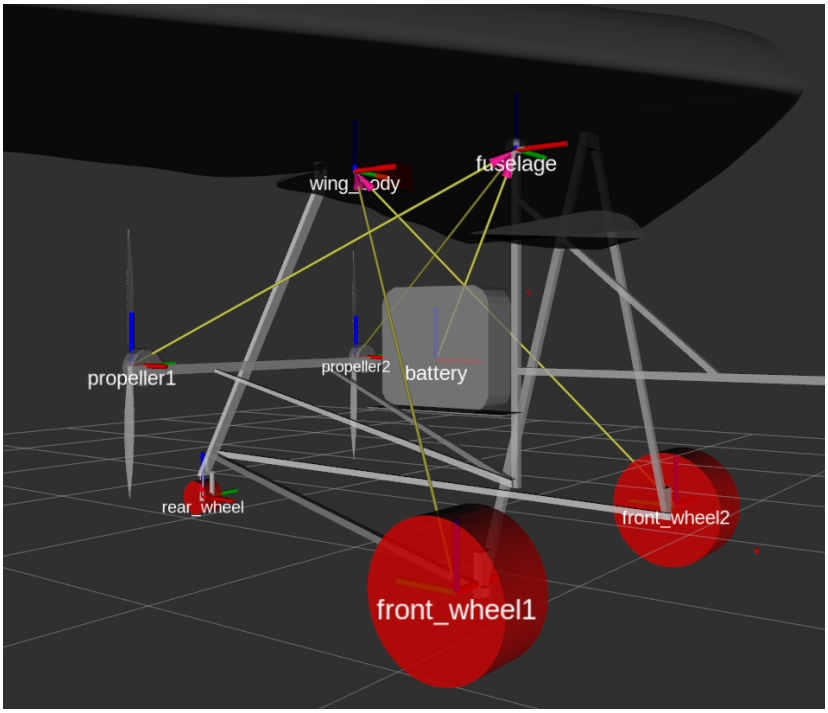


Figure 5.3: Gazebo model visualized with primary components labelled.

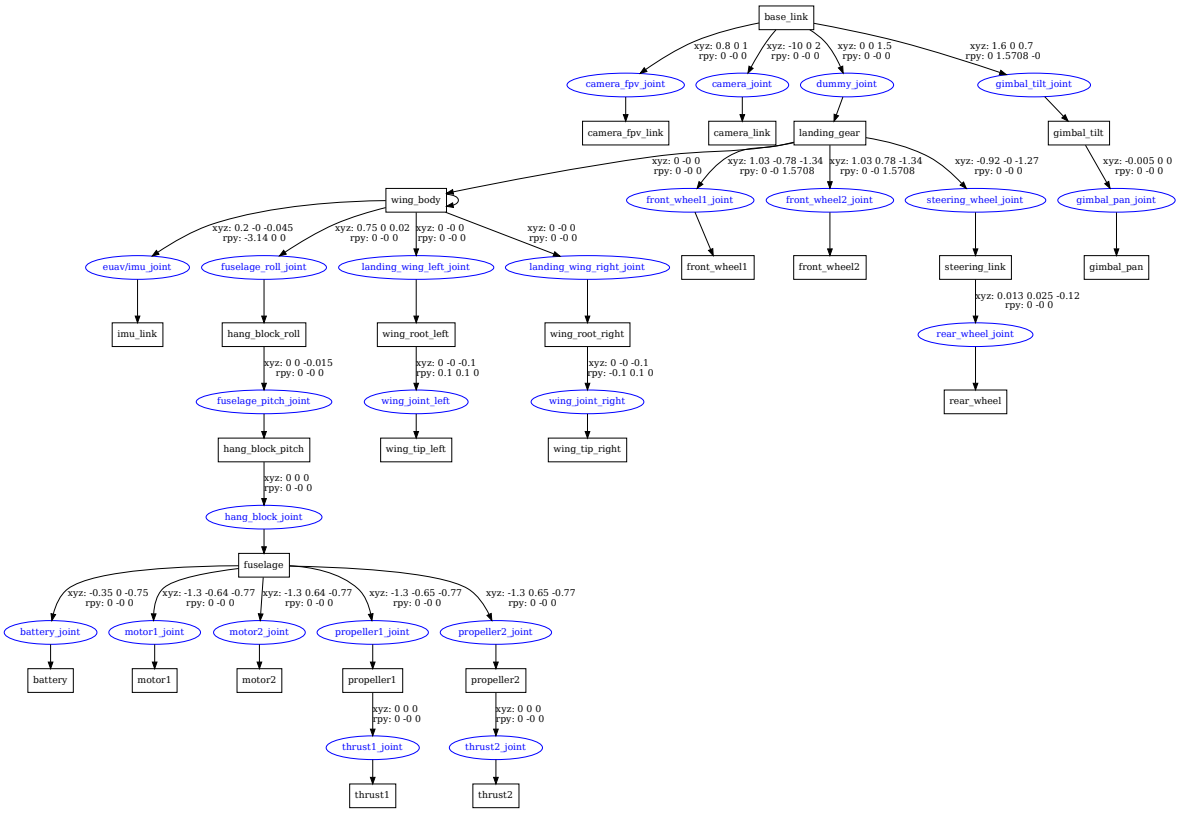


Figure 5.4: Tree diagram representation of the uWSC aircraft model as defined in the standard definition format file that is used by ROS. Black boxes are rigid bodies defined as a combination of a link and a 3D collision volume, Arrows represent injective rigidity maps, XYZ and RPY are pose vectors specify the translation and rotation of each link relative to its rigid parent, and blue ellipses are 1 DoF joint functions. See acronyms definitions in 5.1.

- Wing tips have smaller angle of attack (AoA), while the wing root has a larger AoA.
- Wing tips generate lift as well as pitch down moment.
- Wing root generates lift as well as pitch up moment.
- When the aircraft is pitching up, the wing root tends to stall. The pitch up moment from the wing root will decrease while the pitch down moment from the wing tip will increase, at which point the aircraft will tend to pitch down.
- When the aircraft is pitching down, the wing tip will have negative AoA angle and the pitch down moment from the lift will decrease. At this point, the wing root will have smaller but positive AoA so the pitch up moment will increase, then the aircraft will tend to pitch up.

The approach taken to model correct lateral dynamic stability features of the flexible wing is summarized as follows:

- Wing tips have a dihedral angle (upward inclination of surface in lateral axis), while wing roots have anhedral angle (downward inclination of surface in lateral axis).
- Wing tips generate lift as well as an inward rolling moment.
- The outward rolling moment generated by the wing root, while small, must be accounted for since the approximation of the flexible wing with a rigid geometry does not inherently capture the mitigating effects of wing flexing.
- When tuning the lateral aerodynamic performance, more dihedral angle on wingtips and less anhedral angle on wing roots will produce a better lateral stability range. Doing the inverse will trade-off stability range for increased performance in flight, namely higher flight speeds with less energy consumption.
- When a roll occurs to one direction, relative deflection of the wing tips occurs, with the lowered tip creating a higher lift than the opposite tip, inducing a stabilizing roll moment. That is, as the port side tip moves downwards, the port side will increase in lift, inducing a restoring moment towards the trimmed state.

The Gazebo wing model is divided into four sections to achieve the desired aerodynamic stability properties. Each half of the wing consist of two sections, named the wing tip and

Table 5.4: Aerodynamic parameters pertaining to wing root and wing tip sections of the uWSC aircraft Gazebo model

Param.	**Wing Root**	**Wing Tip**	**Rationale**	**Notes**
α_0	$-11.9°$	$-20.1°$	C_L considered as AoA at zero lift.	Selected from wing performance specification [239].
$C_{L\alpha}$	2.1	2.1	C_L slope before stall.	Estimated from the shape of the airfoil with XFLR5.
$C_{D\alpha}$	0.263	0.63	C_D slope before stall.	Estimated from the shape of the airfoil with XFLR5.
α_{stall}	28.7°	28.7°	AoA at stall point.	Estimated based on aircraft wing performance spec.
$C_{L\alpha_{stall}}$	-3.88	-2.88	C_L slope after stall.	Estimated based on aircraft wing performance spec.
$C_{D\alpha_{stall}}$	-0.43	-0.43	C_D slope after stall.	Estimated from the shape of the airfoil with XFLR5.
S	4.75 m^2	$2.92m^2$	Area of a single wing section.	Estimated from the wing CAD model.
C_p	0.35 ± 0.05 m	0.035 ± 0.02 m	Centre of aerodynamic pressure.	Dependent on wing incidence angle.

wing root. Aerodynamic properties for the sections are confirmed through analysis using XFLR5 [240], where the prototype's wing is modelled as an MH-82 hang-glider airfoil. The wing roots are attached by a revolute joint to the parent keel link, with very large stiffness and friction parameters to limit nearly all motion while allowing for infinitesimal signed deflections, such that the simulator's physics engine applies independent aerodynamic forces to each section. The same process is repeated to attach the wing tips to the wing roots. The wing roots possess a larger effective AoA compared to the wing tips. Table 5.4 presents the aerodynamic parameters for the wing root and wing tip sections of the uWSC aircraft model.

Aerodynamic modelling of the propellers is more straightforward as matching thrust levels of the prototype aircraft design was the only requirement. Table 5.5 presents the aerodynamic properties for the propellers of the uWSC aircraft model.

Table 5.5: Aerodynamic parameters pertaining to propeller airfoil of the uWSC aircraft Gazebo model

Param.	**Value**	**Rationale**	**Notes**
C_p	0.09 m	A half of the total propeller airfoil length.	Constant C_p is assumed hence its location is about a quarter length of the airfoil since that is the assumed profile for the propeller blades, from the airfoil's leading edge.
Area	$0.04\,\mathrm{m}^2$	Area of a single propeller section.	From prototype CAD model for surface area of the propeller airfoil.
Multiplier	595	Produces simulated thrust force value verified to match experimental thrust test of 150 N per motor at maximum rotation.	Gazebo parameter which maps rotation to a force in axis incident to object rotation.

5.3 Flight benchmarking results

The flight performance of the uWSC aircraft model is benchmarked against real-time flight data from thirteen real-world human-piloted weight-shift aircraft missions were provided by the industry partner. The flights were performed by a Cosmos Phase 3 ultralight fuselage equipped with a La Mouette IPSOS 19 flexible wing, [241, 242], equipped with a Navio2 flight controller running ArduPlane purely for data collection purposes [232,234]. The wing is rated for a maximum velocity of 80 km/h, and a total flying weight of 350 kg. Thirteen simulated missions that replicate the real flight trajectories using the uWSC aircraft model are conducted to enable the comparison. The model's quality is validated by comparing simulated flights to real flight data.

The aggregate statistics of all thirteen pairs of flight trajectories are calculated for a set of key flight dynamics parameters; this consist of the measurements for airspeed, wing roll angle, wing pitch angle, wing AoA, side-slip angle, turn rate, and climb rate. Additionally, statistics are calculated for subsets of the flight trajectories along four phases of flight, defined as follows:

- Take-off, representing the initial phase of flight where the aircraft accelerates from the ground to a pre-determined altitude, considered from the start of motion along the runway to the point the aircraft reaches 10 m above ground level and 13 m/s.
- Landing, representing the final phase of flight where the aircraft descends and touches down on the runway, considered from 10 m above ground level and coming to a stop within 25 s.
- Level, representing stable flight where the aircraft maintains a relatively constant attitude, heading, and speed, limited to altitude change of no more than 5 m and a heading change of no more than 40°.
- Turn, representing a maneuver where the aircraft changes its heading to change course, considered as a turn rate of 3 °/s and a heading change of 90° at minimum.

The total flight trajectory data and the defined flight phase section data are interpreted to make direct quantitative comparisons between the real and simulated WSC flights. Note that although the aircraft modelled in the simulation differs from the real aircraft, the purpose is to assess the simulation's ability to match the characteristics of real WSC flights

Table 5.6 compares the statistics of key flight dynamics parameters between the real flights and their corresponding simulated flights. Several observations are drawn from the statistical results. Simulated airspeeds are generally lower than real ones, except during take-off where they align closely. Notable differences emerge in roll during turns and pitch values in general flights, with real data showing higher values than simulated. The large difference in the wing pitch mean is attributed to the different trim conditions for each aircraft. The mean AoA during take-off in real flights is considerably lower than simulations, due to the standard pilot practice of performing a rapid climb out with a large nose-up maneuver after reaching sufficient speed, where as the programmed take-off is gradual by comparison. Landing and take-off in simulation is performed on perfectly flat terrain, and no wind is present in the simulation. Side-slip values during landing indicate a slight drift in real flights absent in simulations. Additionally, real-world flight data often exhibits higher variability, suggesting that real flights are influenced by more unpredictable factors than the simulations capture, such as wind conditions, turbulence, and unregulated control actions of the human pilot. This indicates a need for refinements in the simulation model to better represent actual flight conditions. However, while there are areas of discrepancy, notably in parameters like airspeed and pitch, in several instances such as airspeed during landing and side-slip in level flight, the simulated values closely align with the real-world data. The AoA values in general and level flight phases also reflect a remarkable similarity between

Table 5.6: Comparison of key flight dynamics mean and standard deviation for the real and simulated flight trajectories over all thirteen provided samples.

	Aggregate				**Take-off**				**Landing**				**Level**				**Turn**			
	Real		**Simulated**		**Real**		**Simulated**		**Real**		**Simulated**		**Real**		**Simulated**		**Real**		**Simulated**	
Parameters	Mean	Sd.	Mean	Sd.	Mean	Sd.	Mean	Sd.	Mean	Sd.	Mean	Sd.	Mean	Sd.	Mean	Sd.	Mean	Sd.	Mean	Sd.
Airspeed [m/s]	17.4	2.4	13.5	2.1	11.4	7.9	11.3	2.8	14.8	4.4	8.6	3.2	17.5	1.5	13.1	2.0	18.0	1.7	14.4	1.8
Roll [°]	1.2	9.3	0.3	5.2	0.8	3.9	0.0	0.5	1.3	4.6	0.1	3.2	0.5	7.1	0.1	3.4	4.9	14.0	2.1	8.6
Pitch [°]	16.2	4.7	0.2	3.4	11.9	9.8	5.7	2.4	10.4	4.9	1.4	3.3	17.0	2.9	-0.2	1.9	16.0	3.8	1.6	3.3
AoA [°]	16.0	3.5	14.7	2.0	8.7	7.4	16.8	1.8	11.2	5.9	18.6	8.1	16.7	2.7	14.4	1.1	16.0	2.7	15.3	2.0
Side-slip [°]	-0.6	5.2	-0.5	9.1	1.3	5.8	0.7	1.6	-3.4	3.9	-0.5	16.4	0.1	4.3	-0.6	7.9	-0.7	5.6	0.6	12.3
Turn rate [°/s]	0.3	4.1	0.2	2.9	-0.2	0.9	0.0	0.0	0.3	1.0	0.8	3.1	-0.1	2.6	0.1	1.6	1.6	6.1	1.2	4.6
Climb rate [m/s]	0.0	1.4	0.0	0.9	0.5	2.4	0.8	0.9	-0.3	1.4	-0.6	0.9	0.0	0.8	0.0	0.5	0.0	1.2	0.0	0.9

real and simulated flights. This showcases the effectiveness of the current simulation model in capturing key aspects of real flight dynamics for WSC aircraft.

The performance of the simulated aircraft is further examined by a point to point comparison for a mission track. Figure 5.5 presents the two dimensional latitude-longitude plane projection of the trajectory as well as the altitude over time for a single mission sample. It is observed that the simulated aircraft presents less fluctuations in altitude in contrast to the real aircraft counterpart.

Figure 5.6 presents a qualitative comparison of the simulated aircraft's ability to match the three-dimensional trajectories of the real aircraft along the four flight phase subcategories. Due to automated flight and simplified environmental conditions in the simulation, it's evident that simulated flights exhibit more consistent and quicker take-off and landing phases compared to the varied and lengthier real-life counterparts. The landing phases for the real flight data presents offsets in the altitude level that suggest altimeter calibration issues. The simulated flights are observed to present accurate recreations of the real flight trajectories in the level and turn phases where the simulated flight controller aims to track the trajectory prescribed by the real flight data, in line with expectations suggested by the calculated statistics. The turns of the simulated aircraft are more gradual due to restrictions imposed on the attitude adjustments of the automatic flight control. Overall, the results suggest that the simulated aircraft provides a good approximation of real-world WSC aircraft flight.

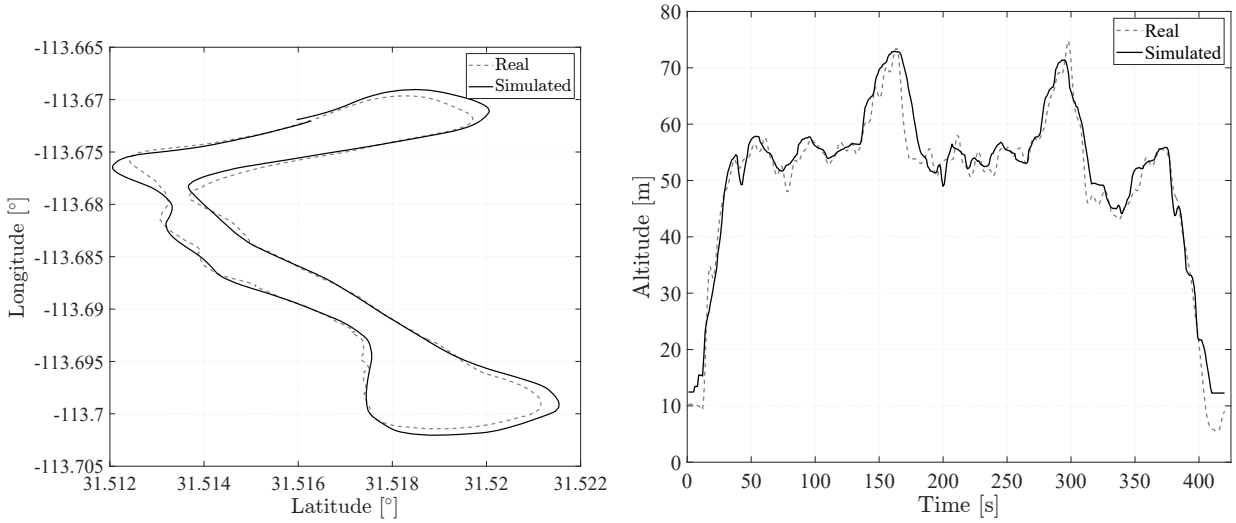
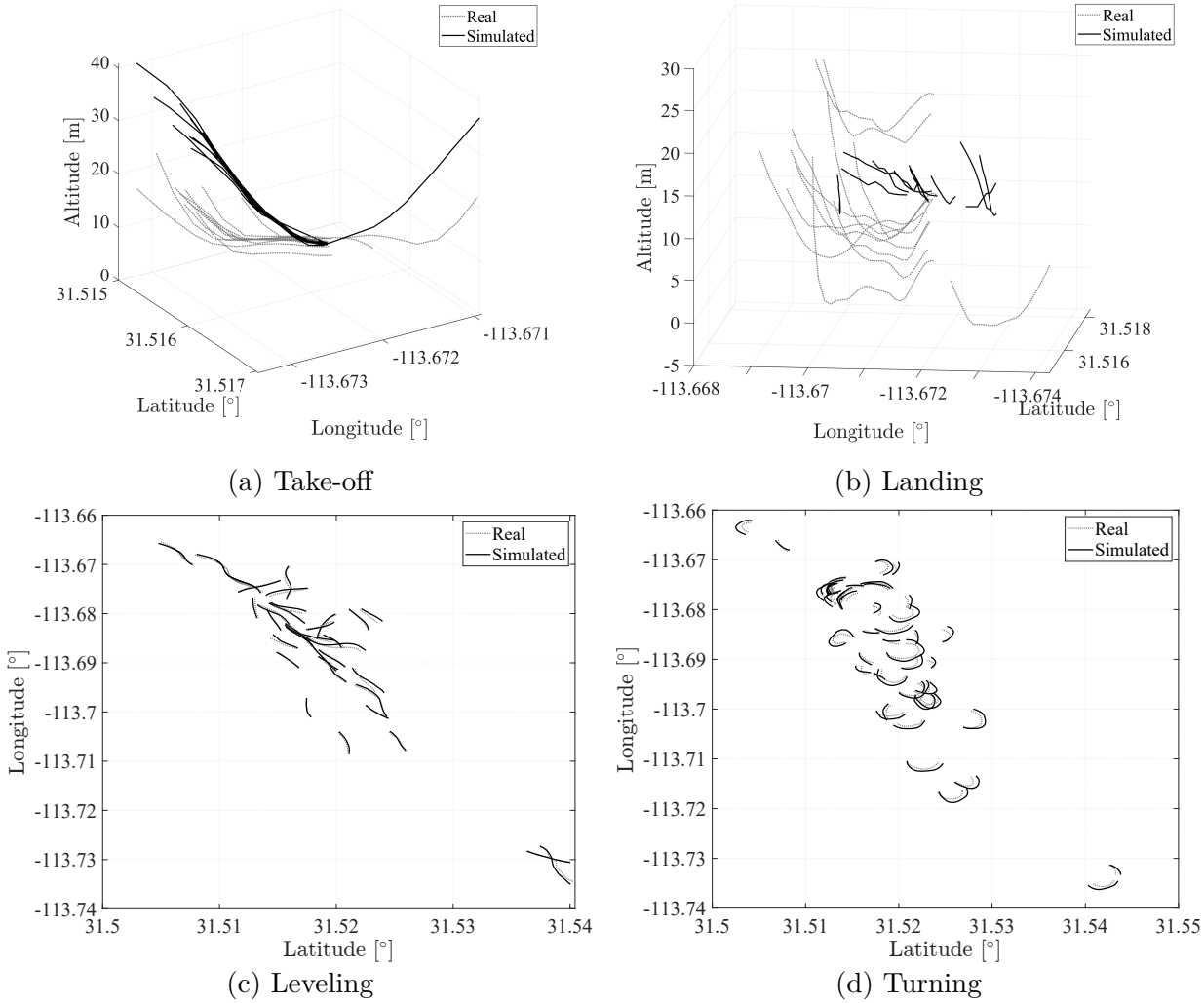


Figure 5.5: (a) Projected ground track and (b) altitude over time for a single mission, comparing simulated and real flight data.



(a) Take-off

(b) Landing

(c) Leveling

(d) Turning

Figure 5.6: Flight trajectories according to four flight phase subcategories, comparing the simulated and real flight data.

5.4 Chapter summary

The uWSC aircraft simulator model offers a means to simulate the dynamics of weight-shift aircraft through Gazebo. The model's fidelity was validated by comparison from real-time data of thirteen weight-shift aircraft missions conducted by the industry partner. While certain flight dynamic parameters, notably airspeeds and wing pitch, showcased differences between the simulated and real flights, several other parameters like the AoA values and side-slip in level flight possess a good resemblance. The simulator's capability for emulating WSC mechanisms in flight is indicative of its utility in bridging higher-level flight controller design with lower-level WSC dynamics. The overall consistency observed in various flight parameters affirms the model's use as a tool for advancing R&D in uncrewed weight-shift aircraft systems.

The uWSC aircraft model has been contributed to the open-source ArduPilot Gazebo repository [40]. Flight data collected from the planned flights of the industry partner's prototype WSC aircraft will be used to improve the simulation aircraft model. In addition, the simulator model will aid in the development of automatic flight controller software specific to weight-shift aircraft. Further refinements of the simulation model will focus on improved approximations for the dynamics of a flexible wing's aeroelastic properties and incorporation of the CDPR piloting mechanism.

Chapter 6

Model-Free Force Control of Cable-Driven Parallel Manipulators for Weight-Shift Aircraft Actuation [4]

6.1 Foreword

This chapter presents the fully developed weight-shift aircraft piloting system. It builds upon the model-free adaptive actor-critic control methods introduced in Chapter 3 and the real online reinforcement learning techniques in Chapter 4 by applying a similar online reinforcement learning strategy to a fully developed weight-shift aircraft piloting system using a cable-driven parallel robot. Notably, the reinforcement learning solution solves the optimal trajectory tracking control problem entirely online using a dynamical structure that combines reference input and tracking error data. It is adapted from the paper titled "Model-Free Force Control of Cable-Driven Parallel Manipulators for Weight-Shift Aircraft Actuation", that was published in the IEEE Transactions on Instrumentation and Measurement journal, in December of 2023. A copy of the original paper is included in Appendix A.4.

This work employs a model-free control approach to overcome the uncertainty induced by the full parametrization of the WSC flexible wing aircraft and CDPR dynamics. Instead of relying on a model, the method directly learns from input-output data. Although basic approximations and hardware tests guide this process, the control strategy does not require a mathematical representation of the system. Thus, model-free control complements model-based methods. It offers a practical solution approach to controlling weight-shift aircraft without being hindered by parameter uncertainties and linearization sensitivities that a pure model-based approach would contend with.

The WSC piloting system is composed of a cable-driven parallel robot (CDPR) integrated with adaptive force control based on reinforcement learning. The weight-shift action has no knowledge of the fuselage state as it manipulates the wing. Coordinated force (torque) control for the set of parallel actuators is determined via the optimal policy identified by online learning. The solution is implemented on flight controller hardware with limited computational resources and strict timing constraints, and shown to perform in real-time experiments on a kinematically equivalent surrogate two-body weight-shift mechanism. To the best of my knowledge, there exists no other control system capable of providing electronic fly-by-wire weight-shift flight control, prior to the solution presented herein.

The primary objective that motived this work was to leverage the advantages of CDPRs as a viable weight-shift control system. The research project detailed in this paper was undertaken to ensure that the CDPR solution could deliver on trajectory tracking flight control requirements. The model-free learning control approach presented ensures that automatic weight-shift flight control would operate without a requirement for sophisticated dynamics models. These designs are tested directly on experimental WSC hardware, to prove their suitability for the demands of real-time automatic flight control for systems like the type schematized in Figure 6.2. The results from this work motivated the industry partner to incorporate a similar weight-shift piloting system solution on their experimental aircraft.

This work presents two principal contributions. The first is the advancement in the design of intelligent flight controllers for weight-shift aircraft. The solution presented is an optimal trajectory tracking controller that is adaptive via the online reinforcement learning mechanism. This model-free approach ensures that the controller can perform despite high parameter uncertainty that is inherent to a modelling-based control approaches for flexible wing weight-shift aircraft [13, 19, 142, 144, 150, 152]. The mathematical derivation of the model-free update laws is based off the method presented by Abouheaf et al. in [22]. Section 6.5 presents recent work developed after publication of the original paper

that demonstrates the design employed for adaptive weight-shift flight control in simulated flight.

The second contribution is the application of online learning control to a real-life CDPR hardware implementation. The approach taken in this work employs an adaptive learning critic scheme to coordinate the parallel manipulators and requires no reference inverse kinematics model. By comparison, conventional approaches of CDPR control rely upon offline dynamic trajectory planning and kinematics models for the parallel manipulator coordination problem. The selected method is an action dependent heuristic dynamic programming (ADHDP) scheme, similar to Q-learning, designed for online implementation utilizing minimal computational resources while requiring no knowledge of the system, reference or input dynamics [62, 106, 114].

The chapter is organized as follows: Section 6.2 introduces the weight-shift flight control mechanism that is realized through a cable-driven parallel robot. Section 6.3 introduces the optimal trajectory tracking problem and its solution using an adaptive actor-critic based online reinforcement learning controller design. Section 6.4 presents experimental results where the CDPR equipped with the model-free adaptive force control coordinates the simultaneous torque commands for each parallel manipulator. Section 6.5 presents results using a modified form of the actor-critic algorithm employed to test an adaptive weight-shift flight control in simulation.

6.2 Weight-shift flight control mechanism

Weight-shift aircraft control consists of force inputs provided by a pilot or piloting system to shift the mass of the fuselage relative to the wing, as illustrated in Figures 6.1 and 6.2(a). The two bodies, the wing and fuselage, are connected at the mast of the fuselage and the keel tube of the wing by the hang-block joint. In crewed flight the pilot situated within the fuselage modifies their orientation with respect to the wing by the reaction from applying force to the control bar attached rigidly to the keel. The centre of gravity (CoG) of the total aircraft is a function of the location of these two primary bodies relative to each-other, and is intentionally modified by the pilot for flight maneuvering. This is illustrated by the example of a banking turn maneuver in Figure 6.1.

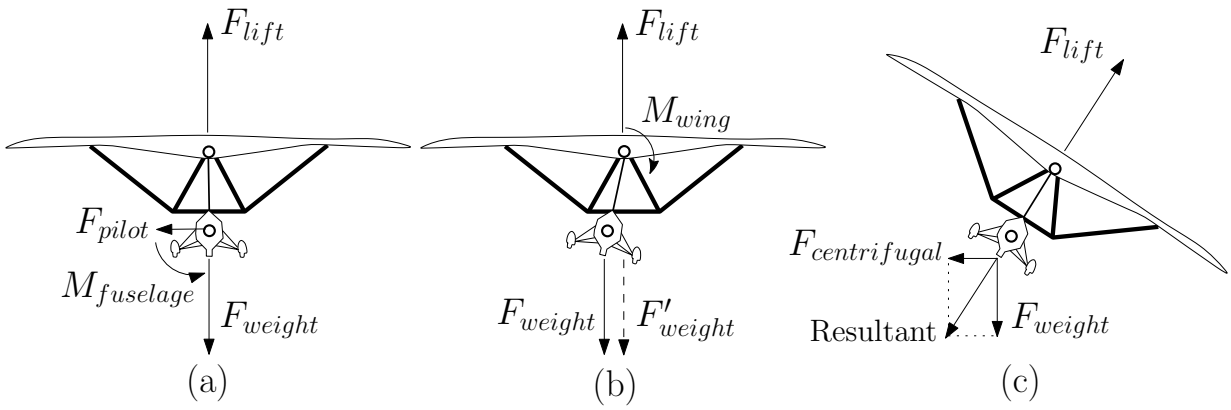


Figure 6.1: Free body diagram sequence illustrating the dynamics of the weight-shift aircraft control during a banking roll: (a) Beginning in balance, (b) pilot induced imbalance produces a weight-shifted dynamic equilibrium, (c) resulting in a banking turn when dynamic balance is restored.

6.2.1 Schematic of the wing-fuselage piloting system

Weight-shift flight control requires that the pilot enacts the correct force input on the control bar of the wing, given the state of the vehicle relative to its flight envelope. When applying force input on the control bar the fuselage body moves relative to the wing body, modifying the centre of gravity for the total aircraft. The fuselage motion, as well as some small motion of the wing due to the initial reaction to the force input, causes a dynamic imbalance as the CoG of the two bodies become misaligned with respect to gravity. An apparent body moment acts on the wing until this balance is restored, causing a change in the lift vector direction, as illustrated in Figure 6.1. In the global reference frame defined with gravity pointing downwards, the wing shifts its orientation due to the weight-shift effect. Dynamic weight balancing is the central principle of weight-shift controlled aircraft maneuvering, allowing the aircraft to perform banking turns or wing pitching maneuvers without conventional aerodynamic control surfaces. The pilot or a piloting system modifies the dynamic balance of the aircraft by applying force against the control bar. The reaction of the pilot force moves the fuselage relative to a frame of reference fixed to the wing system, referred to as the wing reference frame. An additional frame of reference is fixed to the fuselage. Notably, the pilot or piloting system must account for the motion of each body, in both reference frames, to correctly perform the intended weight-shift maneuvers, it therefore has the duty to manage the position, velocity, and acceleration of the fuselage system and the wing system, simultaneously.

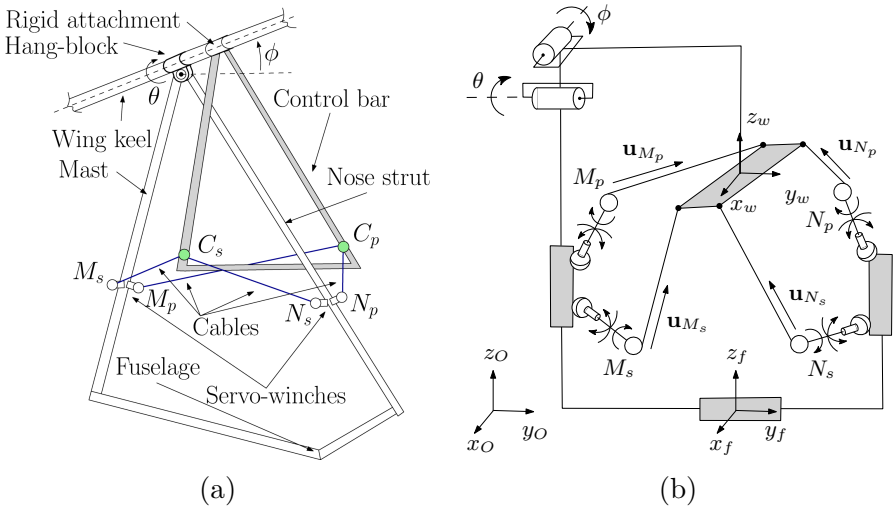


Figure 6.2: Schematic (a) of wing-fuselage system with attached CDPR piloting system; (b) the corresponding kinematic diagram.

A weight-shift aircraft can be schematized by considering the wing and fuselage as two pinned bodies allowed to rotate with respect to each-other in pitch and roll along the hang-block joint. Figure 6.2(a) presents a schematic for the wing-fuselage system with the CDPR piloting system attached. The hang-block is equivalent to two attached revolute joints that rotate about two mutually perpendicular axes. The wing body includes the control bar structure, as it is rigidly attached to the wing keel. The fuselage is supported by the wing during flight and is allowed to pitch and roll along the rotation axes of the hang-block. The rotation of the fuselage about the wing keel has a typical allowed roll rotation range of $\theta = \pm 20^\circ$ and a pitch rotation range of $\phi = -10^\circ$ to 15°. The fuselage is constrained in yaw relative to the wing. The wing has no motion constraints and moves freely within six degrees of freedom (DoF) during flight.

6.2.2 Description of the cable-driven parallel robot

The objective of this work is to design a control mechanism for autonomous flight of flexible wing weight-shift aircraft. The cable-driven parallel robot system replaces the human pilot, allowing for fly-by-wire based weight-shift aircraft control. The CDPR consist of four servo-winches that are mounted at four locations along the mast and nose strut on the fuselage frame, i.e. the body that hangs beneath the wing structure. Each servo-winch is attached

to the fuselage by a spherical joint, such that the cables maintain alignment with the exit of the winch-spools for any configuration of the robot by reorienting the servo-winches along the tension vector of the cables. Figure 6.2(b) presents a kinematic diagram of the cable-driven parallel manipulator mechanism. The fuselage is held at a fixed distance from the hang-block, and rotates freely about two axes of rotation, in pitch and roll, relative to the wing frame of reference.

The servo-winches actuate the fuselage by retracting (and releasing) flexible cables that have their ends attached to points on the control bar. Since the control bar is rigidly attached to the wing, any force applied to it is transmitted to the wing. Hence, depending on which servo-winches are actively pulling, the fuselage will move relative to the wing and control bar structure. Therefore the servo-winches control the motion of the fuselage via the parallel tension force vectors.

The weight-shift mechanism allows the fuselage to rotate with two DoF about the wing keel. By using four cables, the robot can be completely restrained for any pose through the selection of a sufficient tension distribution among the four servo-winches. When a cable is under tension, it mechanically transmits the torque produced by its own servo-winch on the control bar in line with the cable alignment. Since the cables can apply only tension forces, intelligent sequencing of servo torque commands is necessary to control the motion of the fuselage.

6.3 Online model-free solution for the optimal trajectory tracking problem

A model-free trajectory tracking controller for the weight-shift action of the cable-driven parallel robot is designed to overcome high parameter uncertainty and nonlinearity inherent to the system's dynamics. Conventional solution approaches would require knowledge of the system's dynamics to determine the optimal gains offline, backward in time, to apply them in the real time implementation. For linear systems with quadratic performance index, this requires one to explicitly solve the Riccati algebraic equation for the gains [82]. Instead, an online ADP approach is employed, which is implemented in forward time and requires only online measurement of the system's trajectory. Given the system's uncertain and nonlinear dynamics, employing the ADP technique known as ADHDP is appropriate [88, 98]. This method leverages online reinforcement learning to refine estimates for the

agent's performance, and approximate the optimal control policy. This solution treats the system dynamics as unknown, deriving all information from measured trajectories and agent actions.

6.3.1 Optimal trajectory tracking scheme by online model-free ADHDP

The objective is to define a value function structure that enables the learning agent to observe and be responsive to both the tracking error dynamics and to the reference trajectory dynamics, referred to as trajectory stabilization. The aim is to generate an additional control component equivalent to the feedforward signal inherent to classical trajectory tracking. The modified augmented state vector (MASV) ζ_k is defined as the stack of error dynamics and reference trajectory dynamics,

$$\zeta_k = \begin{bmatrix} e_k \\ (e_k - e_{k-1}) \cdot \Delta_t^{-1} \\ n^{-1}\Delta_t \sum_{i=n}^{\infty}(e_i) \\ r_k \\ (r_k - r_{k-1}) \cdot \Delta_t^{-1} \\ (r_k - 2r_{k-1} + r_{k-2}) \cdot \Delta_t^{-2} \end{bmatrix} \tag{6.1}$$

where $e_k = x_k - r_k$, is the tracking error measurement, x_k is the state measurement, r_k is the reference trajectory input, and Δ_t is the time step associated with the measurement interval duration. The objective performance is selected to evaluate system performance across three objectives, namely tracking error minimization, anticipate the reference trajectory dynamics, and minimize the control energy. This defines the value function as

$$V_k = \frac{1}{2}\sum_{i=k}^{\infty} \gamma^{i-k}[\zeta_k^T Q \zeta_k + u_k^T R u_k]$$

, where u_k is the control, and Q and R are weight matrices that are designer selected to determine the learning agent's trade-off for each of the different objectives, and the trade off between control energy/effort and tracking accuracy [62]. Moreover, $\gamma \in (0, 1)$ is the discount factor, and V cannot decrease over successive time steps, i.e. $V_{k+1} - V_k \geq 0$. The summand is the stage cost function, and will be referred to as U_k later on. The discount factor ensures that the learning process does not over value future performance. If the control u_k causes the value function to reach some finite value while stabilizing the system

and is finite, it is considered admissible, but not necessarily optimal. The optimal control is obtained when the value function is minimized according to the deterministic Bellman equation

$$V_k^\star = \min_{u_k}[\zeta_k^T Q \zeta_k + u_k^T R u_k] + \gamma V_{k+1}^\star.$$

Solving explicitly for the optical control gives $u_{k+1}^\star = -\frac{1}{2}\gamma R^{-1} B^T \nabla V_{k+1}^\star$, where B is the control input map.

6.3.2 Actor-critic solution for online value function approximation

The actor-critic solution employed to learn the model-free control online and in real-time for the trajectory stabilization objective is detailed. The Bellman optimality equation acts as a consistency condition, ensuring that the candidate value function follows the optimality principle, converging to the correct desired performance. Two single layer neural networks are trained over time by gradient descent based adaptation laws to approximate the optimal value function (critic unit) and the corresponding controller (actor unit).

The temporal difference (TD) error is zero if Bellman's equation is satisfied, and therefore it is used to quantify the divergence from optimality. In the specific case, the TD error is given by

$$\epsilon_{\text{TD}} = -V_k + \frac{1}{2}[\zeta_k^T Q \zeta_k + u_k^T R u_k] + \gamma V_{k+1}.$$

The TD error is calculated in a value iteration reinforcement learning scheme to find the optimal control and corresponding value function. A value function approximation (VFA) scheme consisting of a projection along neural network basis functions is used. Define the approximate value function as $V_k \approx \hat{V}_k(z_k) = z_k^T \Omega_c z_k/2$, where $z_k = \begin{bmatrix}\zeta_k & u_k\end{bmatrix}^T$ is the data to be used in the approximation method, $\zeta_k \in \mathbb{R}^n$, $u_k \in \mathbb{R}^m$, and $\Omega_c \in \mathbb{R}^{(n+m)\times(n+m)}$ is a symmetric positive definite weight matrix that must be determined. The initial Ω_c^0 is selected as a random symmetric positive definite with small values such that the initial weight adaptation is more gradual and overall numerical stability is improved during the learning process.

To obtain the optimal control, consider the partition

$$\Omega_c = \begin{bmatrix} \Omega_{\zeta\zeta} & \Omega_{\zeta u} \\ \Omega_{u\zeta} & \Omega_{uu} \end{bmatrix},$$

where $\Omega_{\zeta\zeta} \in \mathbb{R}^{n\times n}$, $\Omega_{\zeta u} = \Omega_{u\zeta}^T$, $\Omega_{\zeta u} \in \mathbb{R}^{n\times m}$, and $\Omega_{uu} \in \mathbb{R}^{m\times m}$. Should the optimal $\Omega_c^\star$ be known, the corresponding optimizing control is given by $u_k^\star = \Omega_\pi^\star \zeta_k = -\Omega_{uu}^{\star -1}\Omega_{u\zeta}^\star \zeta_k$, where $\Omega_\pi^\star$ is the policy determined by the optimality condition. During the learning process, a probing white noise is added to the control input signal to ensure that the persistent excitation condition is satisfied [62, 120]. When the learning algorithm converges to the optimal policy, the probing noise is turned off and the learning is halted.

Algorithm 3 Online value iteration with actor-critic structure

1: Initialize state measurements x_{0-2}, reference trajectory states r_{0-2} critic Ω_c^0 as random symmetric P.D. with small values , and actor Ω_a^0.
2: Calculate ζ_0 using (6.1)
3: **begin learning process**
4: **repeat**
5: Update ζ_k using (6.1)
6: Calculate u_k using (6.5), apply to system
7: Allow one time step to pass, $k \leftarrow k+1$
8: Update ζ_{k+1} using (6.1)
9: Calculate TD error ϵ_{TD} using (6.2)
10: Apply critic update law (6.3) to find Ω_c^{j+1}
11: Apply actor update law (6.4) to find Ω_a^{j+1}
12: **until** actor gains have converged and satisfactory trajectory tracking performance is achieved
13: **halt learning process**
14: Calculate further control input with the optimal control $u_k^\star = \Omega_a^\star \zeta_k$

An adaptive critic scheme is employed to identify the critic weight matrix Ω_c. The critic neural network weights are evaluated by calculating the temporal difference of the value function estimate

$$\epsilon_{\text{TD}} = -\frac{1}{2} z_k^T \Omega_c z_k + U_k + \gamma \frac{1}{2} z_{k+1}^T \Omega_c z_{k+1} \tag{6.2}$$

With each iteration step, the critic weights are updated to minimize the square of the TD error, resulting in the adaptive update

$$\Omega_c^{j+1} = \Omega_c^j - \alpha_c \epsilon_{TD} z_k z_k^T \tag{6.3}$$

where α_c is the critic learning rate. The index j tracks the version of Ω_c. The critic weights Ω_c could be used to calculate the optimal policy Ω_π directly as $\Omega_\pi^\star \zeta_k = -\Omega_{uu}^{\star -1}\Omega_{u\zeta}^\star \zeta_k$. However, this approach is not advised as the critic weights have the potential to rapidly

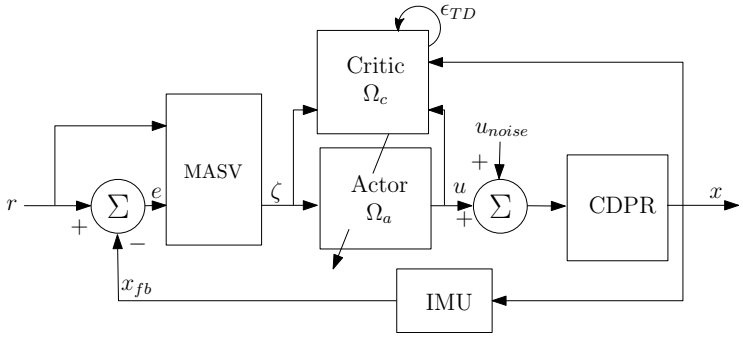


Figure 6.3: Trajectory stabilizing actor-critic model-free controller.

change during the online adaptation. Since the critic is merely found by VFA, this can very well result in consistently poor control policy. A second adaptive actor neural network Ω_a is introduced with actor weights updated according to the adaptive update law

$$\Omega_a^{j+1} = \Omega_a^j - \alpha_a(\Omega_a^j \zeta_k - (-\Omega_{uu}^{j+1})^{-1}\Omega_{u\zeta}^{j+1}\zeta_k)\zeta_k^T, \tag{6.4}$$

where α_a is the actor learning rate. The corresponding optimal control is now given by

$$u_k = \Omega_a^\star \zeta_k \tag{6.5}$$

and $\Omega_a^\star$ is the feedback policy found after convergence of the actor neural network adaptation. The actor-critic adaptation laws provide a means to approximate the trajectory stabilized tracking value function online and in real time, and in turn provides the optimal control solution. Algorithm 3 summarizes the online value iteration actor-critic approach that is used. Figure 6.3 provides the complementary control system diagram.

6.3.3 Implementation of weight-shift control on the CDPR with multi-agent learning

As discussed in Section 6.2, the goal of WSC is to control the attitude of the wing ϕ_w and θ_w, denoting the wing pitch and roll respectively. During in-flight operations the required performance criteria for the pitch control action may be different than that of the roll control action. The CDPR introduced in Section 6.2.2 applies force input to the fuselage by sending simultaneous torque commands across the set of servo-winch actuators to manipulate the wing attitude. Separate commands are necessary to provide the required flexibility for pitch and roll flight controls, hence two separate learning agents are

Table 6.1: Torque division logic, marks indicate the servos activated for the corresponding attitude correction.

	$u^{(1)}: +\phi_w$	$u^{(1)}: -\phi_w$	$u^{(2)}: +\theta_w$	$u^{(2)}: -\theta_w$
M_s	✓	...	✓	...
M_p	✓	...	...	✓
N_s	...	✓	✓	...
N_p	...	✓	...	✓

implemented using Algorithm 3. Each agent is provided the commanded reference trajectory and IMU measurement for a single element of the wing attitude, i.e. $r_k^{(1)} = \phi_w^{cmd}$, $r_k^{(2)} = \theta_w^{cmd}$ and $x_k^{(1)} = \phi_w$, $x_k^{(2)} = \theta_w$, and produces its own control input, denoted as $u_k^{(1)}$ and $u_k^{(2)}$. The control inputs are then mixed according to a simple heuristic that depends on the arrangement of the servo-winches. The multi-agent system diagram is presented in Figure 6.4.

The arrangement of the servo-winch mounts is used to select a heuristic for the relationship between the active servo-winch and the corresponding state change for ϕ_w, θ_w. As illustrated in Figure 6.2(a), the actuators M_s, M_p, N_s, N_p are positioned relative to the end effector attachment points C_s, C_p. The heuristic logic is summarized by Table 6.1. For example, a demanded pitch increase $+\phi_w$ with no roll action implies that the first agent acts, $u_k^{(1)}$ is increased and the demanded torque magnitude is partitioned between M_s and M_p, while the second agent does not act, i.e. $u_k^{(2)}$ does not change. For a second example where a simultaneous demanded pitch increase $+\phi_w$ and a demanded roll decrease $-\theta_w$ occurs, the first agent acts by increasing $u_k^{(1)}$ and the torque is divided between M_s and M_p, while the second agent acts by decreasing $u_k^{(2)}$ and the torque magnitude is divided between M_p and N_p. In the given second example the net demanded torque for M_p is applied as it is being shared by both agents simultaneously. The experimental results within the following section will illustrate the interplay inherent to the parallel actuator system.

6.4 Experimental results and discussion

The proposed model-free learning control is validated experimentally using real flight controller hardware with limited computational resources and strict control update constraints. A prototype CDPR mounted to a surrogate weight-shift mechanism tracks reference wing

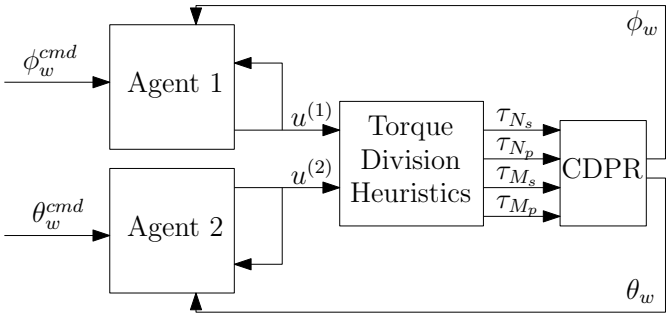


Figure 6.4: Multi-agent learning controller for the cable-driven parallel robot. The agents share the set of servomotors to actuate the weight-shift mechanism.

trajectories ϕ_w, θ_w by shifting the fuselage weight. The selected maneuvers represent repeated S-turns of a real weight-shift aircraft, a standard training maneuver as informed by the industry partner's test pilot [8].

6.4.1 Real-time learning control with weight-shift mechanism

The experimental platform shown in Figure 6.5 reproduces the kinematics to the weight-shift controlled aircraft illustrated in Figure 6.2. The wing analogue is composed of a longitudinal member representing the wing's keel, with a triangle frame rigidly attached to the keel that represents the pilot control bar. The weight-shift fuselage is suspended from the wing keel by a two DoF mechanical joint that is analogous to the hang-block. The wing keel rotates relative to hang point in a range of $\pm 8°$ in pitch and $\pm 10°$ in roll, corresponding to the fuselage displacement relative to the keel of $\pm 20°$ in roll and $-10°$ to $15°$. The control of the platform is actuated by the CDPR. Actuation is provided by four Dynamixel XM-430 servomotors [238] with integrated cable winches. These manipulators are mounted to different positions along the fuselage as illustrated in Figure 6.2(a), and have the other end of their cables attached at positions on the control bar. The servomotors receive current input commands at a rate of 50 Hz, i.e. control update interval $\Delta_t = 20\,\text{ms}$. The adaptation for both critic and actor neural networks is computed within this control update interval directly on the flight controller hardware according to (6.3) and (6.4).

The hardware used is similar to the experimental set-up in Section 4.3. Each hardware device and supporting software is either open-source or has publicly available documentation, [234, 243, 244]. Navio2 provides redundant orientation estimates through the com-

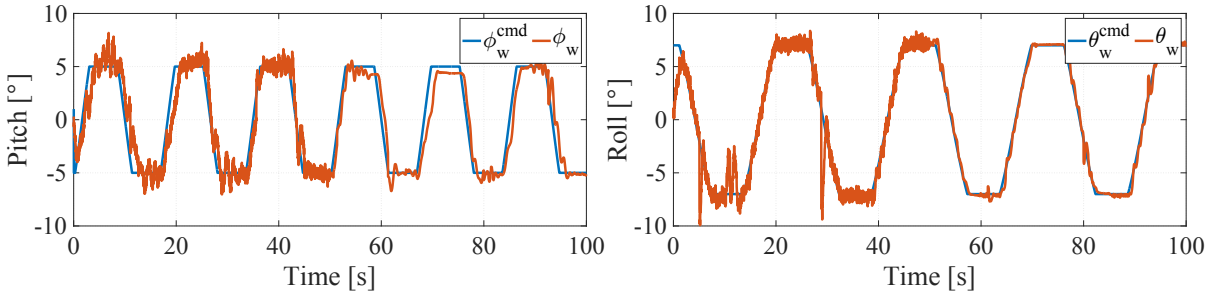
Figure 6.5: Experimental weight-shift control mock-up used to emulate the weight-shift effect of the wing controlled by the CDPR. The wing keel and control bar (blue) are suspended at the hang-block, and are rotated by weight-shift of the fuselage (yellow) by forces on the parallel cables (green).

Table 6.2: Learning controller experimental parameters.

Parameter	Value	Description
Q	$10 \cdot I_6$	state weighting
R	1	control weighting
$\Omega_a^{(i)0}$	$\left[20, 20, 20, 10, 10, 10\right]$	initial actor
$\Omega_c^{(i)0}$	$U(0.001)_{7\times7}$, $\Omega_c^0 > 0$	initial critic
u_{noise}	$N(0, 0.5)$	probing noise
Δ_t	20 ms	update interval
α_c	0.01	critic learning rate
α_a	0.001	actor learning rate
γ	0.999	discount factor

bination of sensor readings from either the main on-board InvenSense MPU-9250 9 DoF Inertial Measurement Unit (IMU) [236] or the secondary Microelectronics LSM9DS1 9 DoF IMU [237]. Each IMU has a three-axis magnetometer, three-axis gyroscope, and three-axis accelerometer. The IMU signals are filtered by a programmable digital low pass filter, which is routed to the Raspberry Pi 3. The total root mean square noise of the MPU-9250 used for inertial measurement feedback is provided to be 0.1 °/s. The measurements data are filtered by a low-pass filter with a cut-off frequency of 250 Hz. The resulting angular position measurements of the wing orientation are sampled at 20 Hz by the online model-free control.

The learning agents determine the correct control by adjusting theirs actions according to measurements of the past trajectory tracking performance. The trained controller then selects the set of demanded torque commands for all actuators simultaneously. A controller agent is trained for each component of the trajectory. In each experiment the training is halted after 50 s and the probing noise is removed. The performance of the converged control policy is observed for an additional 50 s. The set of parameters in Table 6.2 are held constant across all experiments. U and N denote the uniform and normal distributions.



(a) ϕ_w (b) θ_w

Figure 6.6: Trajectory tracking of wing's pitch and roll for coordinated S-turns.

6.4.2 Tracking of weight-shift s-turn commands

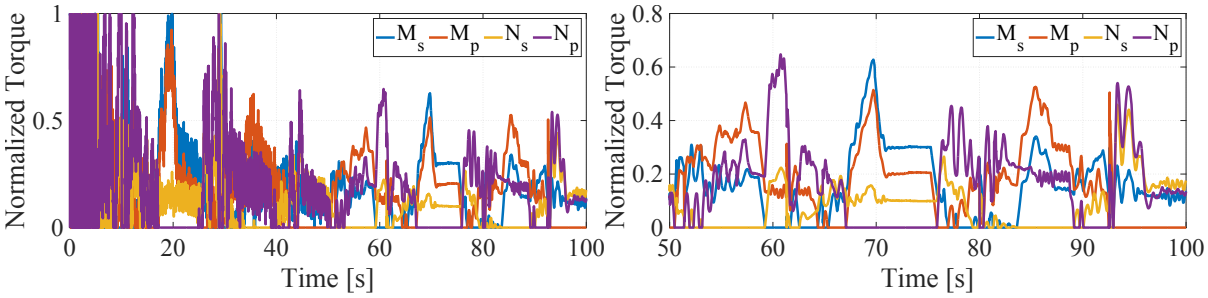
For the first sample result, ϕ_w^{cmd} and θ_w^{cmd} evolve according to the saturating sinusoidal signals that define a repeated weight-shift S-turn, given by

$$\phi_w^{cmd} = \begin{cases} 10\sin(0.06\pi k), & \text{if} |r_k| \leq 5^\circ \\ 5^\circ, & \text{otherwise} \end{cases} \tag{6.6a}$$

$$\theta_w^{cmd} = \begin{cases} 10\sin(0.04\pi k), & \text{if} |r_k| \leq 7^\circ \\ 7^\circ, & \text{otherwise} \end{cases} \tag{6.6b}$$

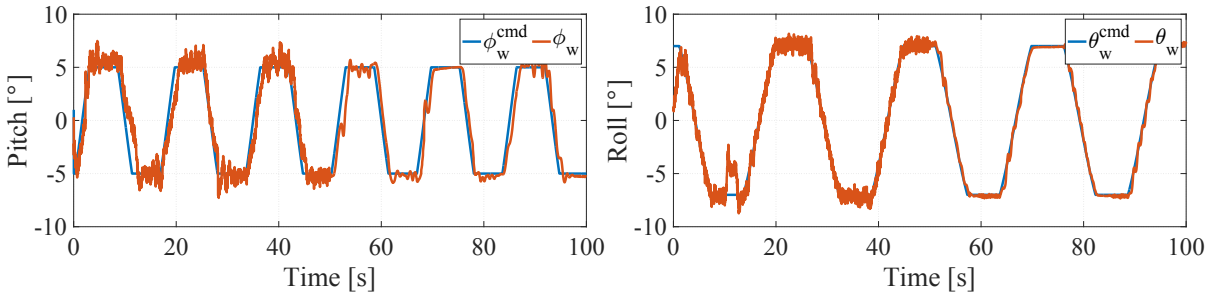
with the goal of demonstrating higher amplitude roll bank commands coordinated with pitch speed control commands. The compound pitch and roll tracking performance of the wing is shown in Figure 6.6. After training is completed, the controller anticipates and tracks the demanded reference trajectory for both axes by application of fuselage weight-shift. When both reference trajectory components change at the same time a transient error is observed as the system adjust each axis in response. After training ends, the actor neural network gains for each agent are $\Omega_a^{(1)} = \left[3.43, 0.26, 9.80, 15.36, 9.03, 12.26\right]$ and $\Omega_a^{(2)} = \left[6.59, 0.14, 11.49, 15.36, 11.04, 13.20\right]$. Figure 6.7 shows the normalized servo torque commands that were generated by the controller during the experiment. It is observed that as the training progresses, the demanded torque variation and amplitudes decrease as the controller learns the optimal policy. Once training ends, the controller generates the correct combination of torque commands for all four servo-motors simultaneously and tracks well the reference trajectory in both axes, as presented in Figure 6.7(b).

To test the adaptability of the learning process, the second experiment applies a sustained disturbance event during training. Disturbances are implemented as random white noise



(a) 0 - 100 s (b) 50 - 100 s, (post-training)

Figure 6.7: Normalized servo torque commands.



(a) ϕ_w (b) θ_w

Figure 6.8: Trajectory tracking with sustained disturbance during learning, beginning at 10 s.

signal sampled from the normal distribution $N(2, 0.5)$ added to the IMU feedback at each time step between 10 s to 15 s, corresponding to 10% of the training period. In the experiment, the disturbances induce a significant tracking error with a magnitude as high as $\pm 5°$. The learning control would rarely converge beyond this magnitude of sustained disturbance. The reference trajectory is kept the same as the prior sample, i.e. evolving according to (6.6). Figure 6.8 shows the tracking performance when the sustained disturbance is present. The controller tracking recovers quickly after the disturbance ends. The trajectory tracking performance achieved after the training ends is unaffected. The disturbance rejection property of the learning algorithm is being tested up to the level of disturbance stated below; further investigation is required to establish if it holds under more extreme events of longer duration or high amplitude, e.g., an unintentional weight offset in the aircraft during flight, or a compromised actuator.

6.4.3 Summary of experimental results

The CDPR system is found to be a viable candidate automatic weight-shift control actuation scheme as it demonstrates the correct kinematic and force interface to match a human pilot performing repeating S-turn maneuvers with coordinated pitch speed input control. The ADHDP technique proves to be viable as a means to implement an online reinforcement learning approach with computationally constrained flight controller hardware. As no model for the system dynamics is known in advance, the adaptive learning agent determines control policy according to the heuristic it is given as presented by Table 6.1. In each experiment, the learning agent identifies an optimal policy in real-time to balance trajectory tracking error minimization, anticipate reference trajectory changes, and minimize the energy used by the servo-torques. Further, it is found that the learning agents can determine an optimal policy even when a sustained disturbance event is present during the learning phase, resulting in a similar set of control gains for both experiments.

6.5 Adaptive flight control simulation

This section presents two recent experimental results developed after the submission of the original paper in this chapter. The previously developed online multi-agent actor-critic controller from the paper are still applied to each problem, but they implement a slightly modified gradient descent update rules for the actor and critic. The results from two problems are examined:

- Section 6.5.2 - An additional CDPR two-body weight-shift experiment .
- Section 6.5.3 - A simulated flight of an updated uWSC model developed with the methodology in Chapter 5, for the first time using the online reinforcement learning for adaptive weight-shift flight control.

Since they are related to the methods presented, they are included with this chapter as an extension of the work. The results demonstrate the practical application of control adaptation through online reinforcement learning in simulated flight conditions.

6.5.1 Modified actor-critic algorithm implementation

This brief note summarizes a modification to the online actor-critic value iteration algorithm that has been studied under ongoing research investigations. The changes are made to the actor and critic update laws. The updated algorithm with the modifications is given in Algorithm 4. The temporal difference error is calculated according to,

$$\epsilon_{\text{TD}} = -\frac{1}{2} z_k^T \Omega_c z_k + U_k + \gamma \frac{1}{2} z_{k+1}^T \Omega_c z_{k+1}, \tag{6.7}$$

the modified critic update law is given as

$$\Omega_c = \Omega_c + \alpha_c \cdot \epsilon_{\text{TD}} \cdot (\gamma \cdot z_{k+1} - z_k) \cdot (\gamma \cdot z_{k+1} - z_k)^T, \tag{6.8}$$

and the modified actor update law is given as

$$\Omega_a = \Omega_a - \alpha_a \cdot (\Omega_a \cdot x_k + \frac{1}{2} \cdot \gamma \cdot g \cdot \Omega_c \cdot z_{k+1}) \cdot x_k^T, \tag{6.9}$$

where g is a heuristic input map. In lieu of a designer chosen heuristic, this is set to weigh the response of all inputs equally,

$$\mathbf{g} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}. \tag{6.10}$$

Algorithm 4 Online value iteration with actor-critic structure with modified update laws

1: Initialize state measurements x_{0-2}, reference trajectory states r_{0-2} critic Ω_c^0 as random symmetric P.D. with small values , and actor Ω_a^0.
2: Calculate ζ_0 using (6.1)
3: **begin learning process**
4: **repeat**
5: Update ζ_k using (6.1)
6: Calculate u_k using (6.5), apply to system
7: Allow one time step to pass, $k \leftarrow k+1$
8: Update ζ_{k+1} using (6.1)
9: Calculate TD error ϵ_{TD} using (6.7)
10: Apply critic update law (6.8) to find Ω_c^{j+1}
11: Apply actor update law (6.9) to find Ω_a^{j+1}
12: **until** actor gains have converged and satisfactory trajectory tracking performance is achieved
13: **halt learning process**
14: Calculate further control input with the optimal control $u_k^\star = \Omega_a^\star \zeta_k$

6.5.2 Experimental result using two-body weight-shift mechanism surrogate with CDPR system

The following results present data from a sample experiment that implements the updated actor-critic algorithm with the CDPR piloting system mock-up. The adaptive trajectory tracking CDPR controller follows a reference trajectory for roll, and pitch is regulated about zero. Figure 6.9 presents the roll trajectory tracking performance throughout the learning process. Three types of regimes can be identified. The initial phase, approximately 0 to 2000 time-steps, shows an improvement regime, where the actor-critic gains change rapidly until the performance reaches good error correction. The second phase, approximately 2000 to 6000 time-steps, shows a stable regime, in which a stable policy is observed. The third phase, approximately 6000 to 10000 time-steps, shows a degenerate regime, where the exploration-induced adaptation leads to a worsened control policy, causing the tracking performance to degrade as a result. From 10000 to 13000 time-steps, a stable regime is restored. From 14000 to 16000 time-steps, a degenerate regime recurs. The experiment is halted after this point.

Figures 6.10(a) and 6.10(b) show the progression of actor neural and critic weights over time. Given the complexity of the critic neural network, with 49 weights in total, it is difficult to assign meaning to their evolution. The critic weights vary greatly in magnitude, as such, the normalized weights are presented. The actor neural network weights map directly to the elements of the augmented state vector ζ, and are therefore interpretable. Figure 6.11(a) captures the accumulated value over the iterations of learning. Figure 6.11(b) illustrates the evolution of temporal difference error during training. It is observed that with sufficient time the VFA begins to predict reward of future state-action as seen during the stable regimes. This indicates that the method was reinforcement learning was functioning well. Figure 6.11(c) demonstrates the temporal difference error with both 100-step and 1500-step moving averages for comparison. Figure 6.11(d) displays log-weighted critic neural network weights with a 100-step moving average.

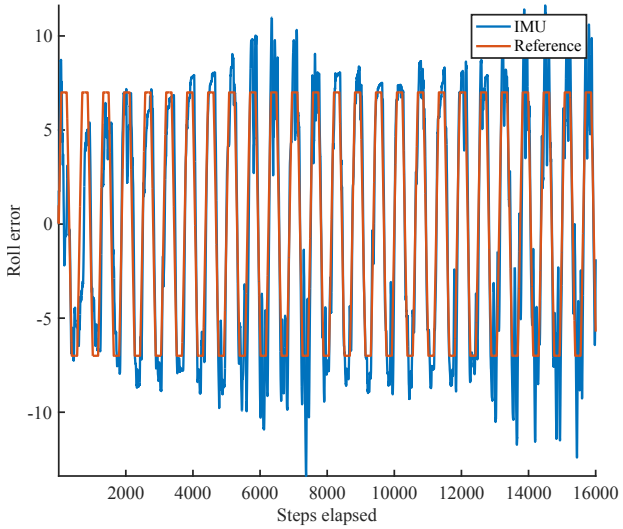
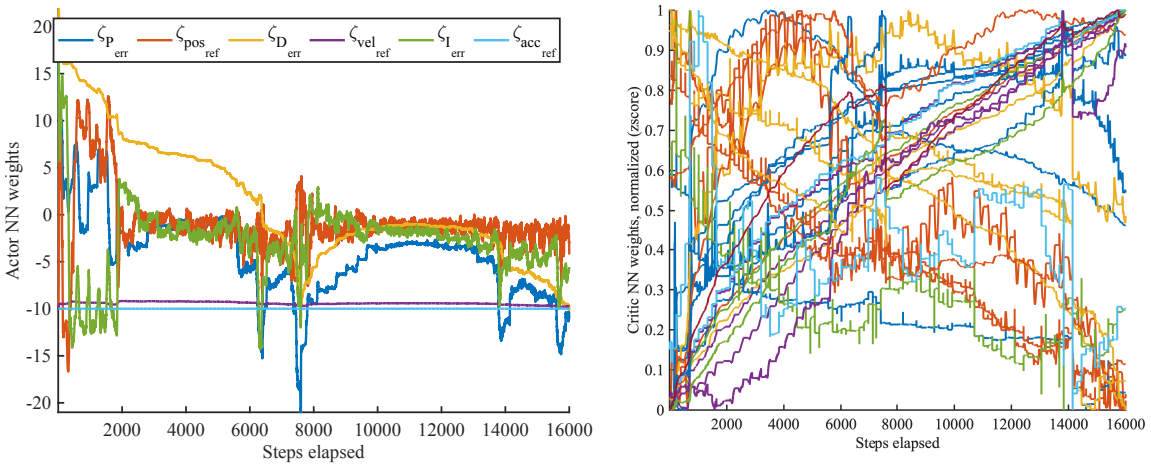


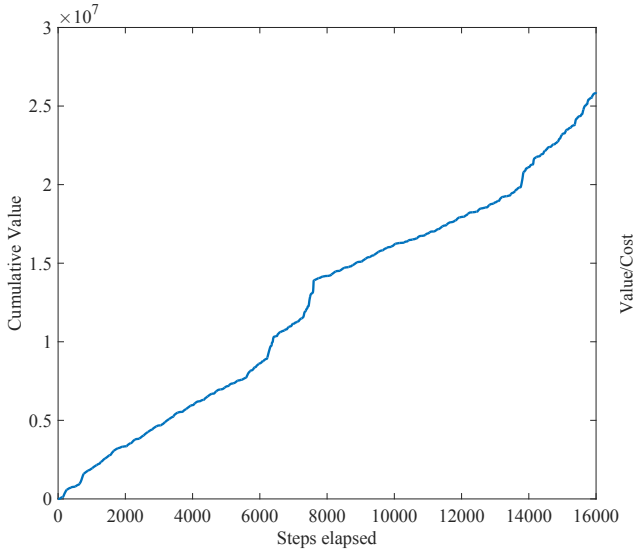
Figure 6.9: Tracking performance during learning. The tracking error decreases as learning progresses.



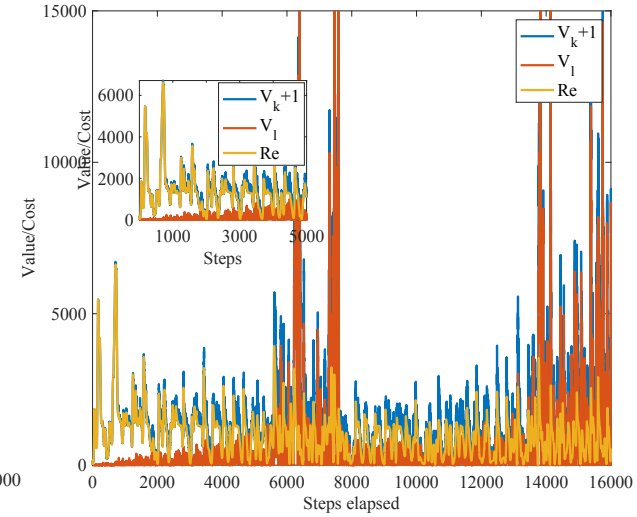
(a) Actor weights

(b) Critic weights (normalized)

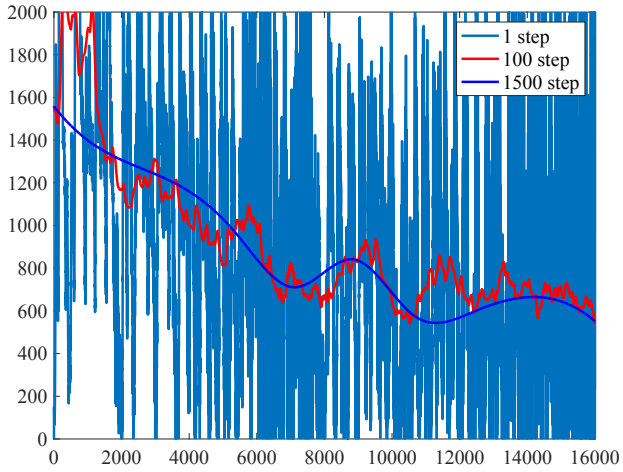
Figure 6.10: (a) Actor neural network weights. (b) Critic neural network weights.



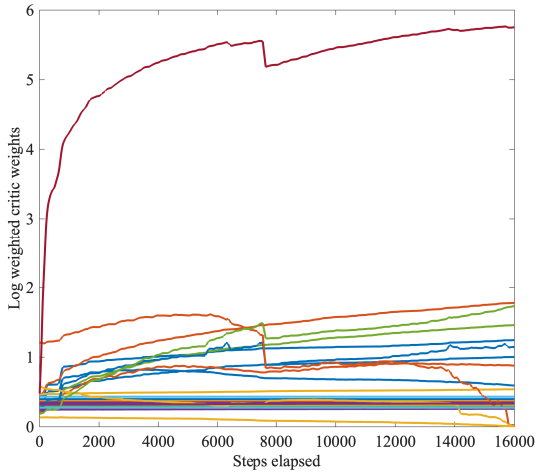
(a) Accumulated value



(b) Value function evolution



(c) TD error



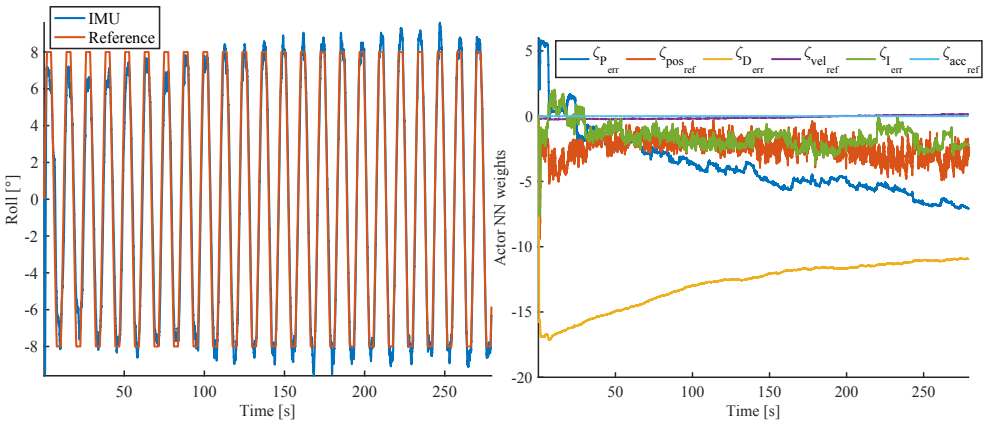
(d) Time average log norm of critic weights

Figure 6.11: (a) Accumulation of value over time. (b) Temporal difference error evolution during training. (c) Temporal difference error, with 100 step and 1500 step moving averages. (d) Critic NN weights, log weighted with 100 step moving average.

6.5.3 Adaptive weight-shift flight control result with uWSC aircraft simulator

The multi-agent learning controller with the modified actor-critic algorithm is implemented directly within ArduPlane, and tested as an adaptive weight-shift flight control for the updated Romaeris uWSC simulator model[1]. In comparison to the CDPR implementations in Section 6.5.2, this implementation generalizes for an entirely different weight-shift actuation mechanism type, i.e. the uWSC model's 2R servo hang-block. The adaptive weight-shift flight control must similarly tracks simultaneous pitch-roll reference trajectories, this time provided by an inexperienced pilot.

The model-free method is applied to the new actuation scheme and must adapt accordingly. An initial pre-training is performed on the ground prior to flight, with the results for the pre-training shown for the wing roll tracking in Figure 6.12, and for the wing pitch tracking in Figure 6.13. To reflect the relative response to control for each weight-shift input, the pitch reference track magnitude is set to be lower than the roll reference track.

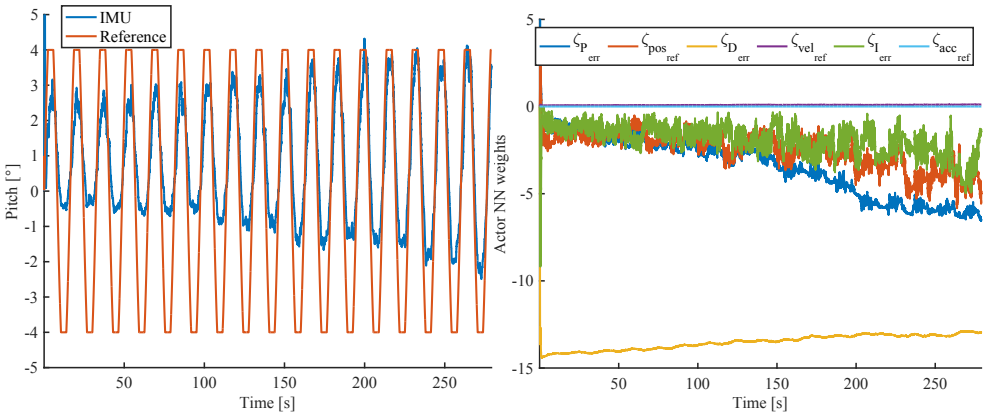


(a) (b)

Figure 6.12: Initial training for ground pre-tune of actuation control mechanism in the roll input, (a) reference track (b) resulting actor weights.

Following the ground pre-training, the actor neural network weights are saved for the pitch and roll controllers. The weight-shift piloting system is then switched to a remotely

[1]I would like to express my sincere gratitude to Trevor Phair and Jazia Djoudad for their support with the new Romaeris uWSC model flight simulation.



(a) (b)

Figure 6.13: Initial training for ground pre-tune of actuation control mechanism in the pitch input, (a) reference track (b) resulting actor weights.

controlled fly-by-wire mode with online reinforcement learning enabled, and is made to take-off and fly a reference circuit that consist of four way points. The inexperienced pilot, still in the loop, observes the flight and inputs reference pitch-roll attitude signals for the intelligent controller to consider. The thrust is directly controlled by the pilot, and is only increased to gain altitude. The adaptive weight-shift piloting system must balance its multiple objectives of reference trajectory tracking stability, tracking error correction, and energy minimization, while having no ability to directly control the thrust. Further, the adaptive action for each control input implicitly considers disturbances induced by the other.

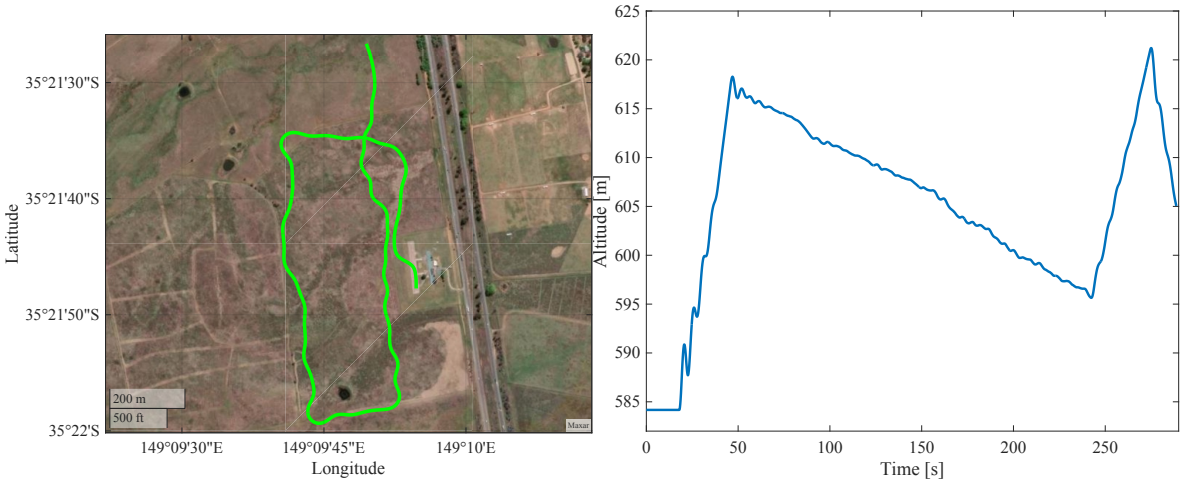
Figure 6.14 presents the projected ground track and altitude over time for the simulated flight. The experiment proceeds as follows:

- After take-off, the aircraft climbs as it approaches the first waypoint at position 35°21′34″ S, 149°09′41″ E around $\approx$ 50 s.
- After this point, the thrust setting is lowered to reduce its effect on the pitch control, and the flight continues.
- The aircraft reaches the second waypoint at position 35°21′59″ S, 149°09′44″ E around $\approx$ 170 s.

- The aircraft reaches the third waypoint at position 35°21′58″ S, 149°09′52″ E around $\approx$ 195 s.
- The aircraft is given more thrust at $\approx$ 250 s to climb away from the ground.
- The aircraft reaches the final waypoint at position 35°21′34″ S, 149°09′49″ E around $\approx$ 275 s, shortly after which the simulated flight control is terminated.

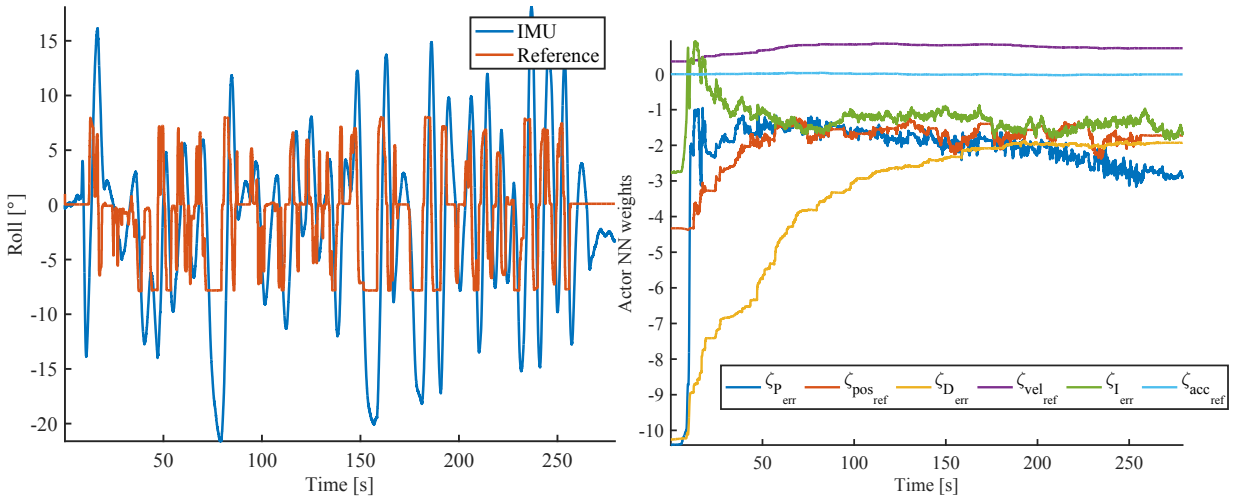
Overall, it is observed that the adaptive learning flight control manages the WSC aircrafts trajectory extraordinarily well despite the pilot's lack of experience. The results for the adaptive optimal trajectory tracking controller are presented for the roll input in Figure 6.15. It is observed that for the roll input, the reference signal is followed by the adaptive piloting system, with some overshoot observed when the roll significantly departs from near trim conditions. In regards to actor-critic adaptation during flight, all control gains required adaptation and balancing, except for the response to the $\zeta_{acc_{ref}}$ signal which remains at zero. The roll input actor neural network appears to decrease response to the $\zeta_{D_{err}}$ signal substantially over time. This would suggest that initially response to error changes was negatively impacting the optimization objectives. The response to the $\zeta_{pos_{vel}}$ signal is adapted, with small magnitude that slightly increases over time, suggesting that roll input quality depends on coordination between the reference trajectory rate and error correction action. Note that the system gains have approximately converge, as the weight of $\zeta_{P_{err}}$ continues to increase in time as the tracking error agent attempts to further optimize for improved error tracking performance. This can lead to numerical instability unless bounds are placed on the error signals.

The adaptive optimal trajectory tracking controller results for the pitch input are presented in Figure 6.16. The pitch reference of the pilot is limited to 10° to 18° to prevent larges inputs that can induce dives or stalls. After 25 s, the adaptive pitch control reduces its sensitivity to error correction signals while maintaining a stronger response to the reference trajectory signal $\zeta_{pos_{ref}}$ and the error rate signal $\zeta_{D_{err}}$. This adjustment results in a sustained offset between the pilot's pitch command and the actual pitch state, which only resolves at the end of the flight when the pilot increases thrust to climb. The adaptive learning agent prioritized trajectory stabilization over error correction in the pitch axis, adjusting its policy to account for the sustained error caused by the lower limit on the reference input. Hence, the piloting system retains a pitch angle that is near the trim condition of 9° despite the active roll inputs observed in Figure 6.15(a).



(a) Projected ground track. (b) Altitude over time.

Figure 6.14: Simulated uWSC flight employing the adaptive actor-critic implementation for flight control.



(a) (b)

Figure 6.15: Adaptive optimal trajectory tracking roll control of weight-shift aircraft via online reinforcement learning, (a) reference track (b) resulting actor weights.

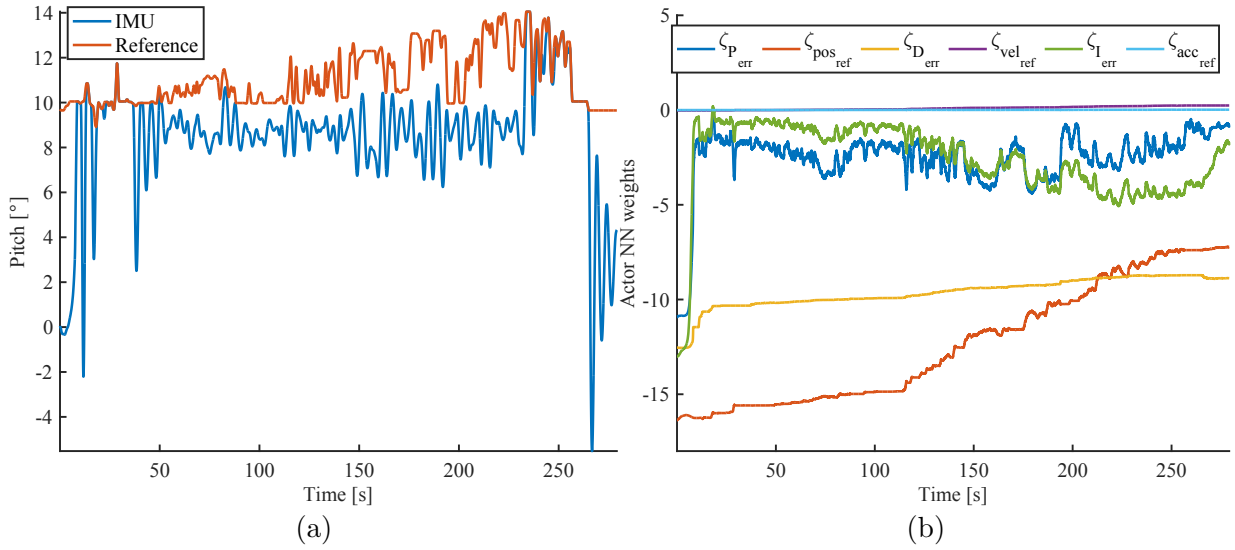


Figure 6.16: Adaptive optimal pitch trajectory tracking control of weight-shift aircraft via online reinforcement learning, (a) reference track (b) resulting actor weights.

6.6 Chapter summary

A solution to automatic weight-shift aircraft control is realized through the application of a cable-driven parallel robot that uses a model-free force controller design that incorporates an adaptive actor-critic based online reinforcement learning algorithm. The ADHDP solution technique allows for an online learning controller to be directly implemented on real flight controller hardware that is computationally limited and requires strict control updates timing. Real-time experiments validate the learning controllers performance by demonstrating its ability to perform weight-shift maneuvers trajectory tracking through the coordination of force applied by the parallel actuators. To the best of my knowledge, this represents the first time a viable automatic weight-shift control mechanism for this class of aircraft has been reported on. The piloting mechanism has full authority over the entire range of WSC, matching the capability of a human pilot.

At the time of writing the thesis, work has been underway with the industry partner to implement the presented solutions in simulated flights and real-life autonomous flight test. Section 6.5 presents an updated iteration of the value iteration algorithm, along with a set of sample experimental results that include a simulated weight-shift flight test where online reinforcement learning provides active control adaptation during flight. The CDPR solution has been tested in experimental flight test by Romaeris, with moderate success, as reported in Chapter 7. In these preliminary tests, the CDPR model-free feedback controller is realized by an error correcting PID design rather than the more sophisticated trajectory tracking actor-critic reinforcement design presented in this work. The initial tuning of the PID gains were determined according to the trained results of the actor weights Ω_u presented in Section 6.4. In the future, the trajectory tracking controller will be implemented along with the adaptive action to enable adaptive tuning for real flight test with the CDPR piloting system. Further, the proposed model-free controller design is applicable to CDPR applications beyond weight-shift aircraft flight control.

Chapter 7

Automatic Weight-Shift Aircraft Control with Experimental Fly-By-Wire Flights [5]

7.1 Foreword

This chapter reports on the experimental validation of automatic flight control for ultralight weight-shift aircraft by application of a novel actuation design that consists of a cable-driven parallel robot. It is adapted from the paper titled "Automatic Weight-Shift Aircraft Control with Experimental Fly-By-Wire Flights" that was submitted to the 2025 CCToMM Symposium on Mechanisms, Machines, and Mechatronics (M3 Symposium), to be held on June 19-20, 2025, in Ottawa, Ontario, Canada. A copy of the paper draft currently under review is included in Appendix A.5.

The CDPR piloting system's hardware and software are proprietary implementations developed by Romaeris that are based on the studies and solutions developed in Chapters 3, 4, 5, and 6. Throughout the thesis work, the research methodology has been shaped to ensure compatibility within the established experimental flight tests regime of the industry partner and adheres to federal flight regulations in Canada, the U.S.A, and Mexico. All test performed by the industry partner prioritize pilot safety and pre-flight validation of the proposed actuation systems. After demonstration of the CDPR's capabilities within a

laboratory setting to produce trajectory tracking for different weight-shifting scenarios, the prototype actuation system is integrated with a human-piloted weight-shift aircraft as an electronic fly-by-wire control for attitude adjustment. The test pilot uses a joystick control scheme to perform several standard weight-shift flight maneuvers. The results validate the CDPR's ability to supplant conventional human-induced weight-shift operations and lay the groundwork for autonomous flight control of weight-shift aircraft.

The chapter is organized as follows: Section 7.2 presents a review of the relevant federal flight regulations under which the industry partner performed experimental flight test. Section 7.3 presents the implementation of the CDPR piloting system with the weight-shift aircraft. Section 7.4 reports on two sample results of positive flight control using the piloting system from flight test provided by the industry partner.

7.2 Federal regulations on ultralight flight

The federal regulations governing ultralight flight in Canada, the United States, and Mexico are included here to raise awareness for the operational, safety, and certification standards that were adhered to during experimental flight test missions. Given that regulations are revised continuously, the regulation summaries provided here refer to their state at the time of flight test in 2022-2023.

7.2.1 Canada

The Canadian Aviation Regulations (CARs), overseen by Transport Canada, defines airworthiness standards for aircraft, including basic ultralight aeroplanes and advanced ultralight aeroplanes, without testing or certifying their design or construction [245]. Basic ultralights must have a maximum takeoff weight of 544 kg and a stall speed of 39 knots or less, while advanced ultralights must adhere to the design standards set by the Light Aircraft Manufacturers of Canada (LAMAC) [246]. Additionally, under CARs Part IV, which addresses Personnel Licensing and Training, pilots of these ultralight aircraft are subject to specific licensing requirements, ensuring they possess the necessary knowledge and skills for safe operation. Operators are mandated to fly during daylight hours, maintain vigilance to avoid other aircraft, and adhere to specific airspace restrictions without creating hazards.

Owners bear full responsibility for maintaining their aircraft in a safe condition. Special Certificates of Airworthiness are available for owner-maintenance, allowing owners who are also licensed pilots to perform and release maintenance beyond elementary work, but these aircraft are restricted to recreational use and cannot be flown in U.S. airspace. Maintenance regulations apply similarly across aircraft, with specific conditions and limitations detailed in the CAR. For Canadian ultralight and amateur-built aeroplanes to operate in the U.S., they must comply with FAA's Special Flight Authorization conditions, and similarly, U.S.-registered Light Sport and Amateur-built aeroplanes need to adhere to Canadian validation terms to operate in Canada. Special Certificates of Airworthiness-Limited cater to aircraft not certified by Transport Canada, including those classified as Light Sport Aircraft in the U.S.A. Owners must obtain a Special Flight Authorization before flying in the U.S., ensuring they meet all required safety and operational standards.

Owners of aircraft with special certificates of airworthiness, whether for owner-maintenance or amateur-built, must adhere to specific maintenance and modification standards. They are required to perform maintenance and modifications in compliance with Canadian Aviation Regulations, using approved data and maintaining accurate records. Certain limitations apply, such as restrictions on modifying owner-maintenance aircraft beyond set conditions, including prohibitions on installing certain types of propellers and floats. Airworthiness directives that address safety concerns must be researched and applied by the owners, ensuring the continued safety and regulatory compliance of their aircraft. Additionally, these aircraft are limited to specific uses, like recreational flying, and cannot be used for commercial purposes or fly in certain airspaces without proper authorization.

LAMAC developed the Design Standards for Advanced Ultralight Aeroplanes (AULA) after Transport Canada introduced the AULA sub-category in 1991, allowing for a specific class of ultralights that exceed the traditional weight limits if they meet certain standards. These standards are outlined in TP10141, now circulated as DS10141, with the latest edition being LAMAC Amendment 003 from November 8, 2004. Manufacturers must provide comprehensive documentation to Transport Canada for an aircraft to be recognized as an AULA, including a Declaration of Compliance, a Specified Maintenance Program, and a Statement of Conformity for each aircraft, which are essential for the initial registration process. The standards ensure that AULAs adhere to stringent design, maintenance, and operational guidelines, supporting the safety and regulatory compliance of advanced ultralight aircraft in Canada.

7.2.2 United States of America

Part 103 of the Federal Aviation Regulations (FARs), overseen by the Federal Aviation Administration (FAA), outlines the operational guidelines for ultralight vehicles in the United States, defining them based on specific criteria related to weight, speed, and usage [247]. An ultralight must be a single-occupant vehicle, used for recreation or sport, without an airworthiness certificate, weighing less than 254 pounds if powered, and having a fuel capacity not exceeding 5 U.S. gallons.

Ultralight vehicles, as defined under FARs Title 14 Part 103, are exempt from the airworthiness certification requirements that are outlined in Part 21 of the same chapter. Specifically, Part 103.7 clearly states that ultralight vehicles do not need to meet the airworthiness certification standards or have certificates of airworthiness that other aircraft are required to obtain under FARs Part 21. This exemption includes the absence of a need for an experimental airworthiness certificate for ultralights, distinguishing them from other aircraft that are used for experimental, educational, or compliance purposes and typically require such certification under Part 21. Operators are mandated to fly during daylight hours, maintain vigilance to avoid other aircraft, and adhere to specific airspace restrictions without creating hazards. The regulations ensure that ultralight activities are conducted safely and responsibly, emphasizing the importance of compliance with designated operational standards to mitigate risks to both the operator and the public.

In addition, the industry partner operated under FARs Title 14 Part 21.190, which mandates light-sport aircraft to obtain a special experimental airworthiness certificate by supplying comprehensive operational and maintenance documentation, along with undergoing an FAA safety inspection [248]. This regulation not only ensures stringent safety standards but also grants these aircraft greater operational flexibility compared to non-certified ultralight vehicles under Part 103.

7.2.3 Mexico

Relative to the U.S.A. and Canada, Mexico has less structured recreational flight regulations [249]. Recreational flight is managed by local authorities and flying communities. The "Reglamento de la Ley de Aviación Civil" outlines regulations relevant to private, recreational, or sport flying, specifically addressing ultralights in the context of Article 14 "Servicio de Transporte Aéreo Privado Comercial" and Article 17 "Servicios Aéreos Espe-

cializados". Article 14 stipulates that commercial private air transport services, including aircraft rental and specialized air services, require authorization and must operate within designated areas. Article 17 expands on specialized air services, encompassing activities such as glider towing, skydiving, and panoramic service, where ultralight aircraft and balloons can be utilized for recreational purposes, highlighting the significance of flight time in service provision.

7.3 Experimental flight test

Flight control solutions are designed within predefined constraints on experimental operation for hardware and software. The industry partner's WSC test pilots actively participate in experiment design in order to comply with essential WSC flight principles. All operations maintain stringent adherence to safety and regulatory standards ensuring that experimental flights are safe [8, 36, 37, 245, 247, 249]. Prototypes are equipped with emergency stops. Actuation systems have firmware-limited actuation authority to ensure safety, where the actuators are limited to a low torque limits such that they can be over-powered by the human pilot. All systems undergoing rigorous ground testing prior to flight test.

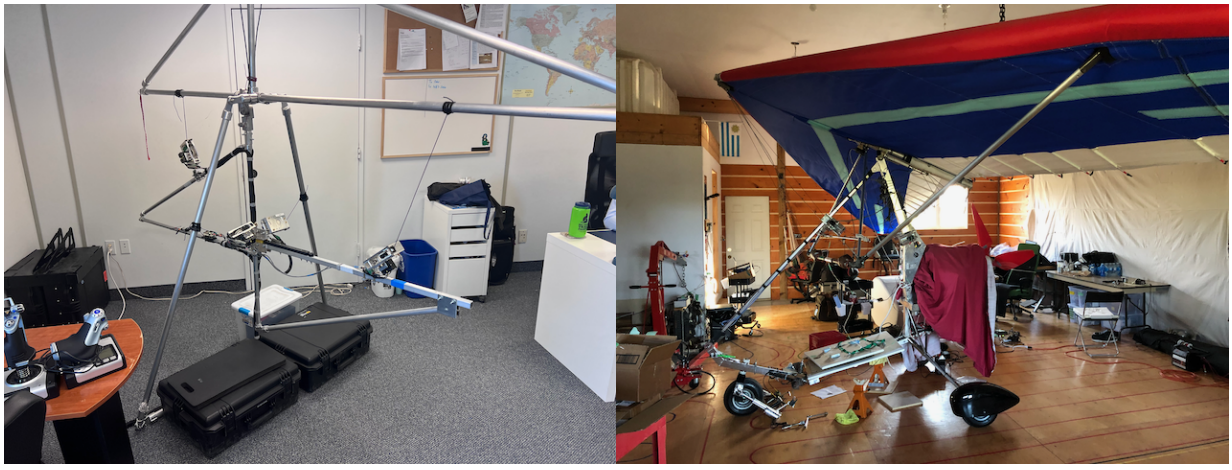
7.3.1 Flight control hardware and instrumentation

The hardware used is similar to the experimental set-up in Section 4.3. The open-source unmanned flight controller software framework Ardupilot is selected as the backbone of the onboard flight data collection in a ride-along mode [234]. ArduPilot is not certified or supported for active controls in manned flights. The industry partner uses separate proprietary sensors and actuation systems that control the aircraft with servo firmware limited authority. ArduPilot provides a complete set of high-level autonomous control functionality, a full suite of sensor support providing state feedback using a Kalman filter based attitude-heading-reference-system, low-level PID controller loops, support for a multitude of different hardware platforms, and extensive community developer support. Given its open-source nature, the software is easily extensible to new unmanned vehicle types, as shown in Chapter 5.

The robotic systems developed followed a progression from mock-ups that mimic the kinematics of WSC aircraft to eventual implementation on WSC aircraft. The experimental

platform shown in Figure 7.1(a) is the most recent iteration of the WSC mockup system. It has identical kinematics to a WSC aircraft prototype. The fuselage is suspended from the wing frame structure by a two DoF mechanical joint that is analogous to the hang-block pivot point of weight-shift aircraft. The control of the platform is actuated by the CDPR. Actuation is provided by four Dynamixel XM-540 servomotors [238] with integrated cable-winches. As illustrated in Figure 6.2, these manipulators are mounted to different positions along the fuselage by spherical joints, and have the other end of their cables attached at positions on the control bar. The servomotors receive torque input commands at a rate of 50 Hz, corresponding to the control update interval $\Delta_t = 20\,\text{ms}$.

The CDPR has been implemented on several experimental aircraft operated by the industry partner. Candidate piloting systems were first tested on the converted WSC aircraft shown in Figure 7.1(b). The WSC capability was demonstrated by a hanging test where the aircraft was hoisted by the lift point. Reference trajectory tracking was tested, along with more essential validation of the hardware and software functions. Once reliability of the piloting system is proven in physical experiments, a program of experimental flight test with a human pilot is conducted. When active, the CDPR system replaces manual inputs of a human pilot in supplying the weight-shifting force. The pilot remains in the loop to direct control input.



(a) Experimental WSC mock-up. (b) CDPR piloting system test.

Figure 7.1: Examples of lab experiment hardware employed to test the CDPR piloting system prior to flight test.

7.3.2 Experimental weight-shift aircraft

Tests of the experimental weight-shift piloting system were performed with the Cosmos Phase 3 ultralight fuselage equipped with a La Mouette IPSOS 19 flexible wing [241, 242]. The wing is rated for a maximum velocity of 80 km/h, and a total flying weight of 350 kg. An image of the test aircraft is presented in Figure 7.2(a). The CDPR actuators are mounted to the chassis and wing down-tubes above the pilot. While this mounting configuration position is sub-optimal in terms of available control authority as the leverage to the hang-block pivot point is reduced, it is an intentional choice that ensures the piloting system out of the way of the human pilot. The servo-motors are further restricted in maximum available torque at the firmware level. Figure 7.2(b) shows the pilot operating the FBW piloting system with a leg mounted joystick.

7.4 Experimental results

A test flight program was launched to validate the CDPR's performance in real-life experimental flight scenarios. After reaching safe operating conditions, the test pilot engages the CDPR piloting system and inputs pitch-roll commands via the joystick. The commands are mapped linearly to a range of torque levels, and mixed according to the torque division heuristic 6.1. A relatively low maximum torque level is implemented at the actuator firmware level to ensure that the human pilot can overpower the CPDR.

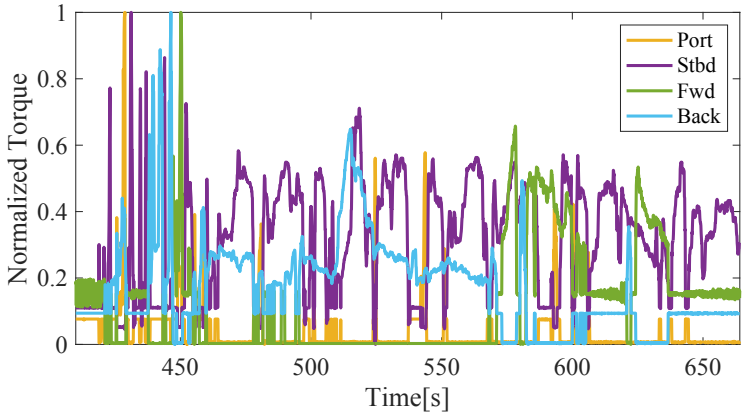
7.4.1 Sample 1: Maiden test of CDPR piloting system

The first test result sample is from the maiden test flight of the piloting system, and shows a fixed altitude starboard-side banking turn maneuver that requires coordinated 2-axis control. Plots for the servo-winch torque commands and the corresponding section of the 2D GPS track from the test flight data log are shown in Figure 7.3. During the maneuver the altitude was held in a horizontal plane between 260 m to 270 m.

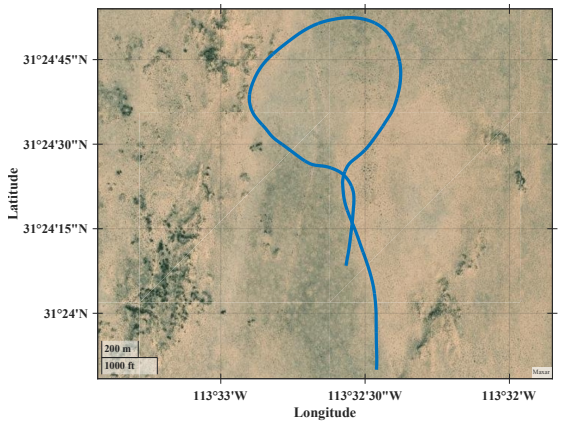
(a) Experimental weight-shift aircraft.

(b) Cockpit view of test pilot sharing his evaluation of piloting system's performance with ground crew.

Figure 7.2: The CDPR piloting system equipped to weight-shift aircraft. When engaged, the CDPR replaces physical pilot control with electronic FBW control provided by the controller.



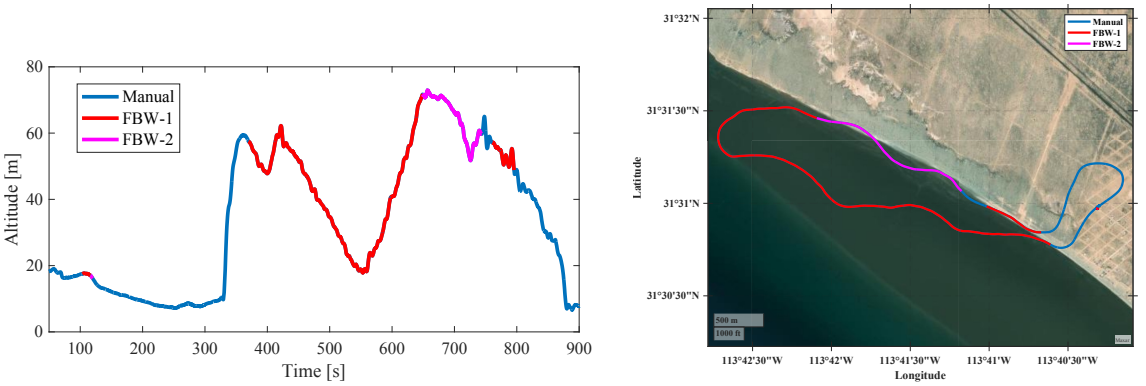
(a) CDPR Piloting system coordinating actuator inputs to induce wing attitude change.



(b) Projected ground track overlaid on satellite map.

(c) 3D visualization of trajectory as it is seen from south vantage point, rendered within Google Earth [250].

Figure 7.3: Flight trajectory of first sample, with piloting system inputs for coordinated turn maneuver.



(a) Altitude over time for automatic flight controls test sample.

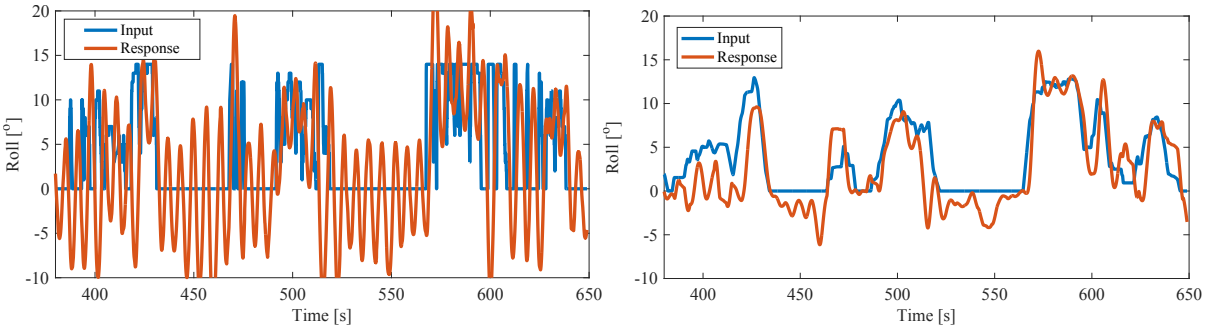
(b) Projected ground track overlaid on satellite map.

Figure 7.4: Flight trajectory of the second sample. The automatic flight controls were active for the track sections labelled FBW-1 and FBW-2.

7.4.2 Sample 2: Automatic flight control with CDPR

The second test sample presents the initial flight test conducted using an automatic feedback based PID flight control to manage the attitude of the weight-shift aircraft, through the CDPR piloting system. The human pilot remains in the loop, monitoring the flight. In contrast with the first sample, the flight controller is responsible for attitude management, while the human pilot controls throttle and provides desired attitude inputs. The controller's PID gains and parallel actuator coordination logic are derived from the model-free reinforcement learning solution presented in 6.

Test pilot notes that average prevailing winds were at 4.7 m/s with a north-east heading of 54° measured from north. The trajectory data for the sample flight test is presented in Figure 7.4. The flight has three control type phases labelled manual, fly-by-wire mode 1 (FBW-1), and fly-by-wire mode 2 (FBW-2) as recorded by the fly-along ArduPilot logger. The actual software implementation of each control type is contained within the proprietary software systems of the industry partner. In manual, the control is a mixture of manual pilot input and direct drive controls with no active feedback control, identical to the first test flight sample. In FBW-1 the feedback controller is commanded by the human pilot to follow roll angles to induce banking turn, pitch is held steady by the CDPR, and throttle is directly controlled by the pilot. FBW-2 is a superset of the FBW-1 mode, where pitch control is additionally employed by the flight controller to manage altitude deviation.



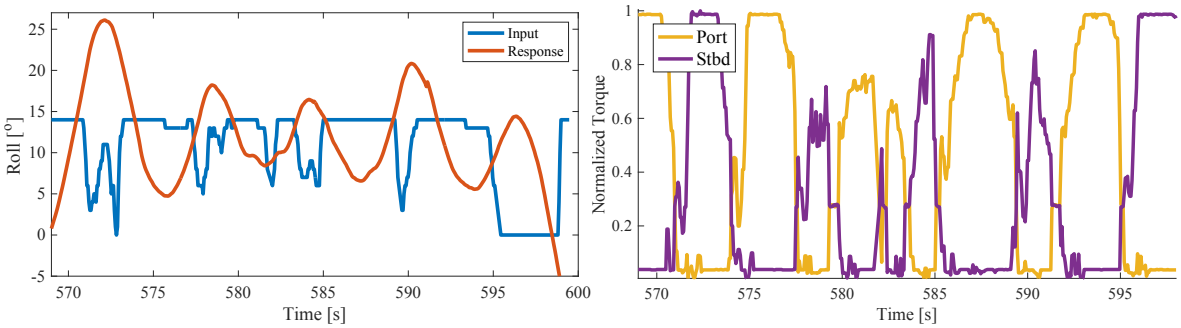
(a) Pilot input and roll response over time.

(b) 2 s moving averages of input and roll.

Figure 7.5: Roll controller input and response of CDPR piloting system while in FBW-1 section of test flight sample.

The flight track as shown in Figure 7.4(a) is better understood when divided into sections as described by the test pilot. The 0 s to 380 s section is in manual control as the pilot climbs out from take-off, gains 60 m of altitude to allow for safe test maneuvers, and performs an in-flight actuation systems check before beginning the automatic controls test. During the 380 s to 570 s section, a brief actuation systems check in FBW-1 is performed to ensure positive control is available, after which the pilot allows the aircraft to descend while inputting turn commands as needed to keep the aircraft on its original heading. During the 570 s to 650 s section, the pilot recovers altitude, and then performs a single large clockwise turn. During the 650 s to 750 s section, the pilot switches to FBW-2 mode to evaluate the pitch-altitude correction response of the controller. The experiment is ended past this time as the pilot switches to manual, disengages the piloting system actuators, and performs landing under human powered input.

In FBW test, the automatic feedback control is interpreting the pilots electronic joystick inputs and correcting for the wing attitude error. Figure 7.5 shows the data captured of the feedback controller's performance. It can be seen in Figure 7.5(a) that the inputs are in the positive direction as the human pilot is commanding the clockwise turn corrections to counteract the prevailing wind that is pushing the aircraft away from the original heading. The aircraft enters a $\pm 10°$ dutch roll oscillation due to the control error correction input. The controller could not attenuate the oscillation as it was not adequately tuned and had limited authority. Figure 7.5(b) shows the result when a 2 s moving average filter is applied to the input and response data, where it is more readily seen that the control performs adequately despite the feedback oscillation.



(a) Pilot input and roll response over time during turn.

(b) The parallel port and starboard side actuators normalized torques over time during the sustained clockwise turn.

Figure 7.6: Roll controller input and response of the CDPR piloting system while in FBW-1 for the large sustained clockwise turn.

The large sustained clockwise turn during the FBW-1 section is worth highlighting for it demonstrates the performance of the piloting system as it operates away from the trim region of the flight envelope. Figure 7.6 presents the input and response performance of the piloting system during the turn. The piloting system detects the aircraft exceeding the imposed safety limit of 15° and performs an immediate automatic corrective action, overriding the pilot's sustained command. The failsafe action repeats 3 times before the aircraft completes the turn. The automatic action of the parallel actuators corresponding to the clockwise turn is shown in Figure 7.6(b). It is clearly observed that the actuators reach the firmware limited saturation point during the turn. This limit prevents the piloting system from performing dangerous weight-shift inputs that have the potential to be catastrophic when near the edge of the flight envelope's stability region.

7.5 Chapter summary

This work reports on the implementation of a novel CDPR based piloting system that enables automatic WSC of flexible wing aircraft. The results validate the piloting system's ability to perform flight maneuvers that would otherwise require physical human-powered input of the pilot. In the cases where the pilot directly manages an open-loop wing attitude control using electronic fly-by-wire based input, positive flight control is achieved. For the cases where basic PID feedback controllers are used and the pilot provides only the reference command, the automatic control somewhat manages the attitude of the wing and maintains stability. However, the feedback controller is not well-tuned and induces significant oscillatory response to the system. Given that the aircraft's dynamics are non-minimum phase, the observed behaviour implies that the controller was overly responsive to the point of increasing the system's sensitivity [32]. Further development of the industry partner's proprietary piloting system designs will incorporate adaptive components to improve the stability provided by the control. Section 6.5.3 presents a result from ongoing research work where an implementation of the adaptive weight-shift flight control employs online reinforcement learning to adapt during simulated flight.

The validation of automatic flight control for weight-shift flexible wing aircraft using a novel CDPR piloting system is a significant milestone that advances the state of the art. The collaboration with the industry partner ensured the feasibility of the proposed actuation and controls design within the context of experimental flight test. Results from test flights demonstrated the CDPR piloting system's ability to control the aircraft attitude effectively. The integration of the CDPR with a manned aircraft and the successful execution of various flight maneuvers validate the system's practical applicability, paving the way for future unmanned weight-shift control aircraft systems.

Chapter 8

Conclusion

8.1 Summary of thesis work

This thesis has advanced the state of the art for the weight-shift control of flexible wing ultralight aircraft by developing and validating a robotic piloting system that successfully emulates human pilot capabilities, paving the way for their broader applications as UAS. A broad set of scientific and technical contributions spanning cable-driven parallel robotics, weight-shift aircraft dynamics modelling and control, and practical design techniques for adaptive critics based online RL were developed in the course of this project. These include:

1. The validation of a prototype robot weight-shift piloting system concept that is experimentally validated to match the capability of a human pilot via electronic fly-by-wire control.
2. A model-free, force-based optimal trajectory tracking controller for cable-driven parallel robots that optimizes coordination of the parallel manipulator torque inputs.
3. Design of adaptive critics algorithms and practical online reinforcement learning techniques, enabling their implementation on computationally-constrained systems requiring low-latency control updates.
4. A technical contribution of an open-source Gazebo model for a unmanned weight-shift control ultralight aircraft that is validated with real WSC flight data to approximate the flight dynamics and stability properties.

By realizing viable solutions to automatic weight-shift flight control, the contributions of this thesis fill a crucial R&D gap, while offering potentially significant improvements towards flight control safety for manned aircraft operations. As shown in Chapter 7, bounded automatic flight control performance is observed when the electronic fly-by-wire piloting system is operated by a human pilot. This lends several key benefits. The piloting system removes the requirement for physically laborious human-powered input, reducing the pilot's potential for fatigue and allowing for enhanced precision in control. This means that the pilot may reassign their focus on the skillful mechanical aspects of weight-shift flight control towards piloting judgement and decision-making, thereby increasing overall operational safety. The piloting system, being robotic in nature, can be designed with adaptive controls to address the stability concerns and non-minimum phase, nonlinear dynamics inherent to this class of aircraft. The robotic system can be equipped with new active safety features, that monitor the flight trajectory in order to enhance the pilot's safety by preventing the aircraft from reaching unstable conditions. An example of this can be seen with the over-bank corrective action feature that is presented in Section 7.4. The preliminary automatic feedback control using static PID gains was able to maintain the stability of the system, however it induced a significant oscillatory response when compared to the open-loop control, due to the non-minimum phase property of WSC.

The inclusion of adaptive control and better controller tuning processes like those shown in simulation results in Chapters 3 and 5, and those shown with experimental results in Chapters 4 and 6 present the solution to resolve the flight controller stability issue entirely while being robust to modelling uncertainty and disturbances. These advancements have the potential to significantly impact weight-shift controlled ultralight aviation by enhancing safety, accessibility, and versatility for recreational flight. The introduction of computer flight controls is expected to boost interest and demand for this aircraft class. Increasing levels of adoption along with a growing pool of pilots and operators will pave the way for new industrial applications that exploit the unique flying properties of these aircraft. As demonstrated by Romaeris, this technology has already enabled the creation of entirely new prototype aircraft designs.

To conclude, the realization of autonomous control systems for unmanned weight-shift aircraft is no longer a question of how it will happen, but a simple matter of when.

8.2 Ongoing and future R&D directions

The section outlines ongoing and future research directions that stem from the primary contributions presented in this thesis. The details given in this section are in progress or exploratory in nature, therefore are not as polished, unlike the main body of the thesis.

8.2.1 Data-driven approaches to learn the dynamics of parallel robots

The originally intended research aim remains to apply data-driven and reinforcement learning techniques to CDPR systems with arbitrary or even unknown parallel actuator placement, such that input maps can be learned online through data feedback alone. Realizing these techniques would eliminate the need for predefined dynamics models, or control action heuristics, such as the one introduced in Chapter 6, which are required to address the problem of parallel actuator coordination. This approach would enable the application of online algorithms to identify optimal control solely through data feedback, thus allowing the system to adapt to changing actuator configurations or other changes during real-time operations, for instance the compromised manipulability near the boundaries of the workspace.

I hypothesize that parallel manipulators, when combined with data-driven techniques, present a significant opportunity as they inherently provide a rich input data space. This richness arises from the interactions among the redundant input components and the system as a whole. This effect has not yet been directly studied in the literature. As a beginning step in this direction, a new method that extends dynamic mode decomposition (DMD) to incorporate the effects of control (DMDc) has been applied [251]. This method provides the ability to extract low-order models from high-dimensional measurement data for complex systems, even when dynamics and modes are influenced by external forcing. Snapshots of observable states and actuation data are used to generate input-output models without need for the underlying governing equations in advance. This method is relatively simple to implement. Variants of optimized DMD have been developed to address bias error from noisy measurements, providing stable and robust probabilistic models useful for forecasting future complex system states [252–254].

Figure 8.1 presents a sample result of CDPR dynamic model identification. The test data is gathered from a CDPR weight-shift control mock-up system shown in Figure 6.5. The test

uses approximately 1000 s of measurements spaced at 20 ms for the mock-up wing's roll and pitch attitude, θ and ϕ, along with the four parallel actuator servo-torque inputs τ_{1-4} that provide the weight-shift input. Let $x = [\theta, \phi]^T$ be the system state, and $u = [\tau_1, \tau_2, \tau_3, \tau_4]^T$ be the control. Application of the DMDc method generates the linear dynamic model given by

$$\begin{bmatrix} \dot{\theta} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} 0.9893 & 0.0011 \\ -0.0014 & 0.9881 \end{bmatrix} \begin{bmatrix} \theta \\ \phi \end{bmatrix} + 10^{-3} \begin{bmatrix} -0.2428 & 0.2851 & -0.2225 & 0.2597 \\ 0.1575 & -0.1794 & -0.3000 & 0.2061 \end{bmatrix} \begin{bmatrix} \tau_1 \\ \tau_2 \\ \tau_3 \\ \tau_4 \end{bmatrix}.$$

The resulting root mean squared error between the linear dynamics model derived states relative to the measurement data is 2.62° for roll, and 2.21° for pitch.

The linear model was validated with an additional 500 s of measurement data, as shown in Figure 8.2. In this case, the previously identified dynamic model is simplify stepped forward using the input time-series to generate the simulated result. The resulting root mean squared error between the linear dynamics model derived states and the measurement data is 2.69° for roll, and 2.30° for pitch.

In the thesis work, in lieu of mathematical models to map from task space to joint space, the CDPR control solutions have required the definition of a heuristic that encodes the basic actuation logic for the parallel manipulators, e.g. Table 6.1. It is postulated that adaptive critics schemes are capable of learning an approximation for the parameters of the inverse Jacobian matrix to map the end effector velocity (or wrench) to the desired joint velocities (or torques) of each manipulator. Reinforcement learning techniques can be combined with partial or heuristic models to learn the underlying kinematic structure of the CDPR. Algorithms that perform sparse regression upon designer selected candidate function libraries to identify nonlinear dynamic models from system data have been combined with reinforcement learning techniques [255–258]. Donge et al. have applied DMD methods to precondition data used by ADP-based online reinforcement learning control techniques for complex multi-agent systems [259, 260]. With this style of approach, the actor-critic algorithm would learn to optimize the performance of the data-driven CDPR system model by minimizing cable tension while tracking desired trajectories accurately in the presence of disturbances or uncertainties. This will eliminate the requirement for pre-defined heuristics, as the map between task space and joint space will be determined online. Recent literature is relevant to this research direction. Lyu et al. use a neural network-based data-driven approach for task-space tracking control of robots with unknown kinematics and dynamics using both offline and online learning approaches is presented in [123]. This

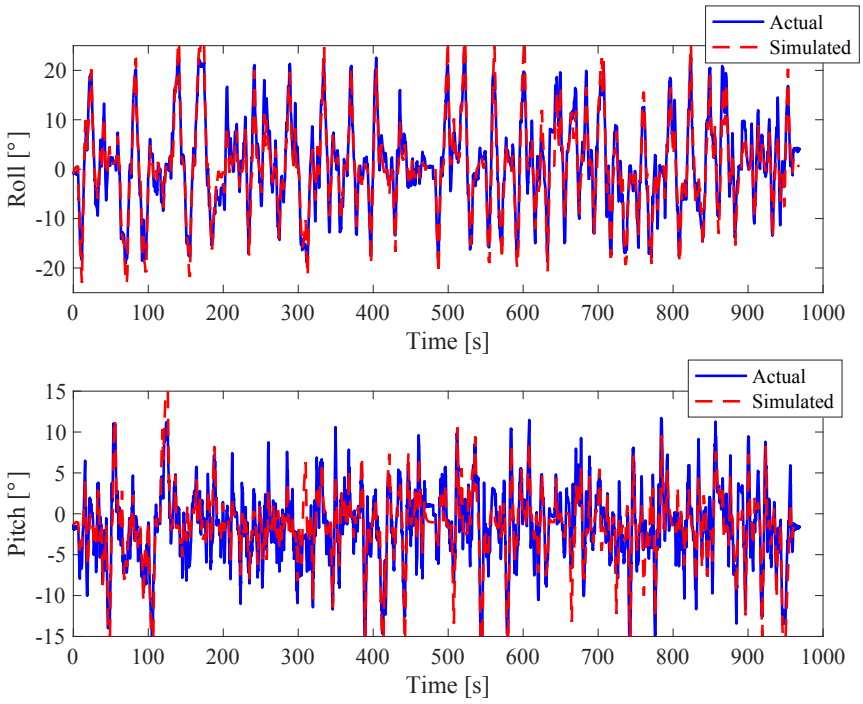


Figure 8.1: Results from DMDc where linear dynamic model is trained with data captured from CDPR.

work demonstrates the control of a serial link robot with an unknown Jacobian matrix by training neural networks to approximate the map between joint space and task space. As an alternative, Xiong et al. use feed-forward neural network models that are trained offline to map the relationship between cable tensions and the corresponding wrench applied on the end effector [197]. In essence, they approximate the inverse kinematic map

$$\tau = \mathbf{J}^{-T} w$$

where J is the forward Jacobian that maps from joint space to task space, τ is the vector of cable-tensions, and w is the vector representing the end-effector wrench applied. They achieve this by training a neural network to identify the relationship that considers the current pose and wrench applied to the resulting cable-tensions,

$$\tau = \hat{\mathbf{J}}^{-T} w \simeq N(w, x)$$

where x is the vector containing the pose data of the end effector, and $N(w, x)$ is the equivalent nonlinear neural network function approximation. The referenced methodology

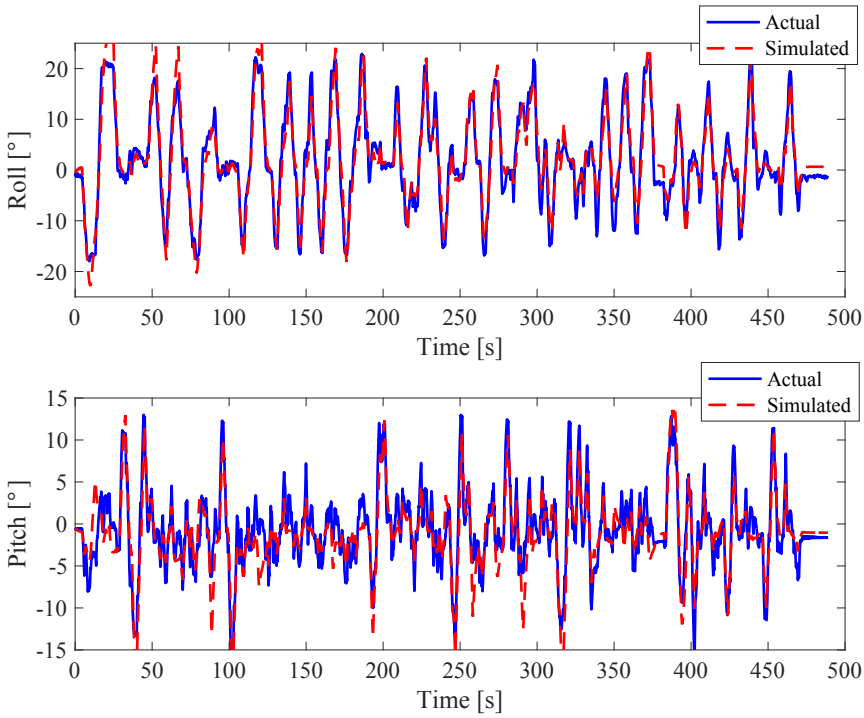
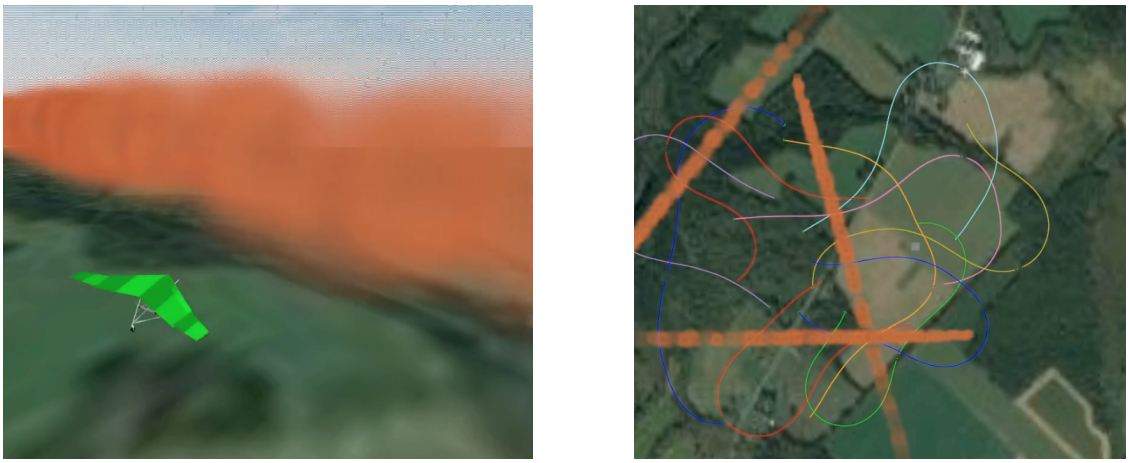


Figure 8.2: Results from DMDc test using trained model against validation data.

utilizes single hidden layer nonlinear neural networks and conducts offline training and validation using simulation with Gazebo. Notably, their PID control scheme does not account for disturbances. Future research would look to extend this approach by developing a model-free control strategy that trains a nonlinear neural network entirely online. This design would map task space commands to joint space while incorporating disturbance rejection capabilities.



(a) (b)

Figure 8.3: Example mission scenarios considered using the simulator methodology. (a) The uWSC model detecting gas emissions. (b) A multi-agent approach for the emissions detection mission.

8.2.2 Realizing new unmanned weight-shift aircraft prototypes

The uWSC simulation model has been used for scenario analysis where airborne multi-agent systems perform gas emissions detection missions, as presented in Figure 8.3. Work has been ongoing with Romaeris to develop a small (10 kg to 25 kg) remotely piloted weight-shift aircraft prototype, with the objective that it can be made autonomous. The uWSC modelling approach detailed in Chapter 5 has already been leveraged as part of the design process to develop new flight dynamics model for candidate designs under consideration. The simulator was used to estimate dynamic balance and flight control properties of the prototype before initial flight tests. The use of the simulator allows for flight controller designs and tuning to be studied prior to real-life experimental flights, improving the likelihood for successful flight tests. Already, the maiden flight of the first unmanned prototype exceeded the expectations of the industry partner, creating the foundation for future research and technology developments. An image of the model in Gazebo simulation, and the set of images of the maiden test flight for the prototype are presented in Figure 8.4. Trajectory data from the first real-world autonomous uWSC flight is presented in Figure 8.5.

(a) Gazebo model for the prototype unmanned weight-shift aircraft.

(b) Images from maiden flight of prototype unmanned WSC aircraft.

Figure 8.4: Simulation and maiden flight images of the prototype unmanned weight-shift aircraft, showcasing the progression from Gazebo modelling to real-world testing. The top image displays the Gazebo model, while the bottom series illustrates a sequence in the first flight, demonstrating the practical application of the design and control developed in the simulation.

(a)

(b)

Figure 8.5: First autonomous uWSC flight, circa November 2024, (a) Waypoint navigation track, (b) An image of the aircraft flying overhead.

References

[1] Mohammed Abouheaf, Nathaniel Mailhot, and Wail Gueaieb. An online reinforcement learning wing-tracking mechanism for flexible wing aircraft. In *Proceedings of the 2019 IEEE International Symposium on Robotic and Sensors Environments (ROSE)*, Ottawa, ON, Canada, 2019.

[2] Nathaniel Mailhot, Mohammed Abouheaf, Wail Gueaieb, and Davide Spinello. Guidance mechanism for flexible wing aircraft using measurement-interfaced machine learning platform. *IEEE Transactions on Instrumentation and Measurement*, 69(7):4637–4648, July 2020.

[3] Nathaniel Mailhot, Teresa de Jesus Krings, Boyan Zhou, Gilmar Tuta Navajas, and Davide Spinello. uWSC Aircraft Simulator: A Gazebo-based model for uncrewed weight-shift control aircraft flight simulation. In *Proceedings of the 2023 IEEE International Symposium on Robotic and Sensors Environments (ROSE)*, Tokyo, Japan, 2023.

[4] Nathaniel Mailhot, Mohammed Abouheaf, and Davide Spinello. Model-free force control of cable-driven parallel manipulators for weight-shift aircraft actuation. *IEEE Transactions on Instrumentation and Measurement*, 73:1–8, December 2023.

[5] Nathaniel Mailhot, Teresa de Jesus Krings, and Davide Spinello. Automatic weight-shift aircraft control with experimental fly-by-wire flights. In *Proceedings of the 2025 CCToMM Symposium on Mechanisms, Machines, and Mechatronics*, Ottawa, ON, Canada, June 2025. submitted.

[6] Guy B. Gratton. The weightshift-controlled microlight aeroplane. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 215(3):147–154, 2001.

[7] Michael J. Kroes and Michael S. Nolan. *Aircraft Basic Science*. McGraw-Hill Education, New York, 8th edition, 2013.

[8] Federal Aviation Administration and Paul Hamilton. *Weight-Shift Control Aircraft Flying Handbook*. Aviation Supplies & Academics, Inc., 2015.

[9] Paul Dees. Hang glider design and performance. In *AIAA Aviation Technology, Integration, and Operations Conference*, Fort Worth, Texas, September 2010.

[10] S. Metzger, W. Junkermann, K. Butterbach-Bahl, H. P. Schmid, and T. Foken. Measuring the 3-D wind vector with a weight-shift microlight aircraft. *Atmospheric Measurement Techniques*, 4(7):1421–1444, July 2011.

[11] Romaeris Corporation. `https://www.romaeris.com`, 2024. Accessed: April 30, 2024.

[12] Mitacs. Artificial intelligence (AI) powered adaptive flight controller for novel unmanned aerial vehicle (UAV) with commercial and humanitarian applications. `https://www.mitacs.ca/en/projects/artificial-intelligence-ai-powered-adaptive-flight-controller-novel-unmanned-aerial-vehicle`, 2019. Accessed: March 6, 2023.

[13] Eric Lanteigne, Justin McLeod, Mayank Vadsola, and Shilong Liu. On the design of structural wing members for an unmanned weight-shift aircraft. *Journal of Unmanned Vehicle Systems*, 8(3):161–171, 2020.

[14] Mitacs. A novel approach to the design of an unmanned aerial vehicle for humanitarian initiatives. `https://www.mitacs.ca/en/projects/novel-approach-design-unmanned-aerial-vehicle-humanitarian-initiatives`, 2016. Accessed: March 6, 2023.

[15] Amit Gaur, Jyoti Budakoti, Chung-Horng Lung, and Alan Redmond. IoT-equipped UAV communications with seamless vertical handover. In *IEEE Conference on Dependable and Secure Computing*, pages 459–465, August 2017.

[16] Charles Vidal, Azzedine Dadouche, Iryna Borshchova, Javad Gholipour, Kris Ellis, Iraj Mantegh, David Orchard, Hali Barber, and Derek Gowanlock. Transport Canada RPAS R&D yearly progress report (2020-2021). Laboratory Technical Report LTR-SMM-2021-0033, National Research Council of Canada, 2021.

[17] Iryna Borshchova and Kristopher Ellis. DAAMSIM: A simulation framework for establishing detect and avoid system requirements. *Drone Systems and Applications*, 10(1):266–286, 2022.

[18] Ilan Kroo. *Aerodynamics, Aeroelasticity and Stability of Hang Gliders*. PhD thesis, Stanford University, Moffett Field, California, 1983.

[19] Michael V. Cook and M. Spottiswoode. Modelling the flight dynamics of the hang glider. *The Aeronautical Journal*, 109(1102):1–20, 2005.

[20] Yoshimasa Ochi. Modeling of flight dynamics and pilot's handling of a hang glider. In *AIAA Modeling and Simulation Technologies Conference*, pages 1758–1776, 2017.

[21] Anthony Pizarro and Bruno Doerwald. Unmanned aerial vehicle with synchronized sensor network, U.S. Patent and Trademark Office US20200192404A1, 2018.

[22] Mohammed Abouheaf, Wail Gueaieb, and Frank L. Lewis. Online model-free reinforcement learning for the automatic control of a flexible wing aircraft. *IET Control Theory & Applications*, 14(1):73–84, 2020.

[23] Mohammed Abouheaf and Wail Gueaieb. Reinforcement learning solution with costate approximation for a flexible wing aircraft. In *IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications*, pages 1–6, June 2018.

[24] Mohammed Abouheaf, Wail Gueaieb, and Frank L. Lewis. Model-Free Gradient-Based Adaptive Learning Controller for an Unmanned Flexible Wing Aircraft. *Robotics*, 7(4):66, December 2018.

[25] Mohammed Abouheaf and Wail Gueaieb. Neurofuzzy Reinforcement Learning Control Schemes for Optimized Dynamical Performance. In *2019 IEEE International Symposium on Robotic and Sensors Environments (ROSE)*, pages 1–7, June 2019.

[26] Mohammed Abouheaf, Wail Gueaieb, and Davide Spinello. Online multi-objective model-independent adaptive tracking mechanism for dynamical systems. *Robotics*, 8(4):82, December 2019.

[27] Mohammed Abouheaf and Wail Gueaieb. Online model-free controller for flexible wing aircraft: a policy iteration-based reinforcement learning approach. *International Journal of Intelligent Robotics and Applications*, 4(1):21–43, March 2020.

[28] Nathaniel Mailhot. Private conversations with Bruno Doerwald, CTO of Romaeris Corporation, and expert weight-shift pilot. Private communication, 2018–2023. Conversations spanned topics related to flexible wing aerodynamics, weight-shift pilot flight training, and mission operations.

[29] Guido De Matteis. Dynamics and control of hang-gliders. In *American Institute of Aeronautics and Astronautics 16th Atmospheric Flight Mechanics Conference*, Boston, MA, August 1989.

[30] Yoshimasa Ochi. Pilot's CG location and attitude control for lateral maneuver of a hang glider. In *30th Congress of the International Council of the Aeronautical Sciences*, Daejeon, Korea, 2016.

[31] Alberto Isidori. A tool for semi-global stabilization of uncertain non-minimum-phase nonlinear systems via output feedback. *IEEE Transactions on Automatic Control*, 45(10):1817–1827, October 2000.

[32] Gunter Stein. Respect the unstable. *IEEE Control Systems Magazine*, 23(4):12–25, August 2003.

[33] Ruggero Frezza, Alessandro Beghi, and Alessandro Saccon. Model predictive for path following with motorcycles: application to the development of the pilot model for virtual prototyping. In *43rd IEEE Conference on Decision and Control (CDC)*, volume 1, pages 767–772, December 2004.

[34] Transport Canada. Recreational aircraft airworthiness. `https://tc.canada.ca/en/aviation/aircraft-airworthiness/recreational-aircraft-airworthiness`, February 2019. Accessed: March 15, 2023.

[35] Transport Canda. Get permission for special drone operations. `https://tc.canada.ca/en/aviation/drone-safety/drone-pilot-licensing/get-permission-special-drone-operations`, oct 2020. Accessed: March 15, 2023.

[36] Transport Canada. Transport Canada Aeronautical Information Manual (TC AIM) - TP 14371. `https://tc.canada.ca/en/aviation/publications/transport-canada-aeronautical-information-manual-tc-aim-tp-14371`, October 2022. Accessed: March 15, 2023.

[37] Federal Aviation Administration. Special Flight Authorizations. `https://www.faa.gov/aircraft/gen_av/ultralights/sfa`. Accessed: March 15, 2023.

[38] Apache License, Version 2.0. `https://www.apache.org/licenses/LICENSE-2.0`. Accessed: March 15, 2023.

[39] The GNU General Public License v3.0 - GNU Project - Free Software Foundation. `https://www.gnu.org/licenses/gpl-3.0.en.html`. Accessed: March 15, 2023.

[40] Nathaniel Mailhot, Teresa de Jesus Krings, Boyan Zhou, Trevor Phair, Gilmar Tuta Navajas, and Davide Spinello. Model for the Romaeris unmanned weight-shift control (uWSC) electric aircraft. `https://github.com/ArduPilot/SITL_Models/blob/master/Gazebo/docs/WeightShiftAircraft.md`, 2024. Accessed: April 1st, 2024.

[41] Brian L. Stevens, Frank L. Lewis, and Eric N. Johnson. *Aircraft Control and Simulation: Dynamics, Controls Design, and Autonomous Systems*. John Wiley & Sons, 1st edition, 2015.

[42] Arthur E. Bryson. Optimal control - 1950 to 1985. *IEEE Control Systems*, 16(3):26–33, 1996.

[43] Qian Wang and Robert F. Stengel. Robust nonlinear flight control of a high-performance aircraft. *IEEE Transactions on Control Systems Technology*, 13(1):15–26, January 2005.

[44] Farid Ahmed-Zaid, Petros Ioannou, Ken Gousman, and Robert Rooney. Accommodation of failures in the F-16 aircraft using adaptive control. *IEEE Control Systems Magazine*, 11(1):73–78, January 1991.

[45] Zachary T. Dydek, Anuradha M. Annaswamy, and Eugene Lavretsky. Adaptive control and the NASA X-15-3 flight revisited. *IEEE Control Systems Magazine*, 30(3):32–48, June 2010.

[46] Richard A. Hyde and Keith Glover. The application of scheduled H_∞ controllers to a VSTOL aircraft. *IEEE Transactions on Automatic Control*, 38(7):1021, July 1993.

[47] Yuri Shtessel, James Buffington, and Siva Banda. Tailless aircraft flight control using multiple time scale reconfigurable sliding modes. *IEEE Transactions on Control Systems Technology*, 10(2):288–296, March 2002.

[48] Ekprasit Promtun and Sridhar Seshagiri. Sliding mode control of pitch-rate of an F-16 aircraft. *IFAC Proceedings Volumes 17th IFAC World Congress*, 41(2):1099–1104, January 2008.

[49] Mirza Tariq Hamayun, Christopher Edwards, and Halim Alwi. Design and analysis of an integral sliding mode fault-tolerant control scheme. *IEEE Transactions on Automatic Control*, 57(7):1783–1789, July 2012.

[50] Halim Alwi and Christopher Edwards. Fault detection and fault-tolerant control of a civil aircraft using a sliding-mode-based scheme. *IEEE Transactions on Control Systems Technology*, 16(3):499, May 2008.

[51] Gang Feng. A survey on analysis and design of model-based fuzzy control systems. *IEEE Transactions on Fuzzy Systems*, 14(5):676–697, October 2006.

[52] Karl J. Åström and Björn Wittenmark. *Adaptive Control*. Dover Publications, 2nd edition, 2013.

[53] Karl J. Åström and Purushotam R. Kumar. Control: A perspective. *Automatica*, 50(1):3–43, January 2014.

[54] James B. Rawlings. Tutorial overview of model predictive control. *IEEE Control Systems Magazine*, 20(3):38–52, June 2000.

[55] Max Schwenzer, Muzaffer Ay, Thomas Bergs, and Dirk Abel. Review on model predictive control: An engineering perspective. *The International Journal of Advanced Manufacturing Technology*, 117(5-6):1327–1349, November 2021.

[56] Thomas George Thuruthel, Egidio Falotico, Federico Renda, and Cecilia Laschi. Model-based reinforcement learning for closed-loop dynamic control of soft robotic manipulators. *IEEE Transactions on Robotics*, 35(1):124–134, February 2019.

[57] Daniel Bruder, Xun Fu, Richard B. Gillespie, Christian D. Remy, and Ram Vasudevan. Data-driven control of soft robots using koopman operator theory. *IEEE Transactions on Robotics*, 37(3):948–961, June 2021.

[58] Shen Yin, Xianwei Li, Huijun Gao, and Okyay Kaynak. Data-based techniques focused on modern industry: An overview. *IEEE Transactions on Industrial Electronics*, 62(1):657–667, January 2015.

[59] Girish V. Chowdhary and Eric N. Johnson. Theory and flight-test validation of a concurrent-learning adaptive controller. *Journal of Guidance, Control, and Dynamics*, 34(2):592–607, March 2011.

[60] Rushikesh Kamalapurkar, Lindsey Andrews, Patrick Walters, and Warren E. Dixon. Model-based reinforcement learning for infinite-horizon approximate optimal tracking. *IEEE Transactions on Neural Networks and Learning Systems*, 28(3):753–758, March 2017.

[61] Zhong-Sheng Hou and Zhuo Wang. From model-based control to data-driven control: Survey, classification and perspective. *Information Sciences*, 235:3–35, June 2013.

[62] Frank L. Lewis, Draguna Vrabie, and Vassilis Syrmos. *Optimal Control*. Wiley, New York, USA, 3rd edition, 2012.

[63] Thomas Chaffre, Julien Moras, Adrien Chan-Hon-Tong, Julien Marzat, Karl Sammut, Gilles Le Chenadec, and Benoit Clement. Learning-based vs model-free adaptive control of a MAV under wind gust. In *19th International Conference on Informatics in Control, Automation and Robotics*, pages 362–385, 2022.

[64] Muhammad Nauman, Wajiha Shireen, and Amir Hussain. Model-free predictive control and its applications. *Energies*, 15(14):5131, January 2022.

[65] Willy K. Wojsznis and Terry L. Blevins. Evaluating PID adaptive techniques for industrial implementation. In *Proceedings of the 2002 American Control Conference*, volume 2, pages 1151–1155, May 2002.

[66] Michel Fliess and Cedric Join. Model-free control. *International Journal of Control*, 86(12):2228–2252, December 2013.

[67] Wanrong Li, Huawei Yuan, Sinan Li, and Jianguo Zhu. A revisit to model-free control. *IEEE Transactions on Power Electronics*, 37(12):14408–14421, December 2022.

[68] Elmira Madadi, Yao Dong, and Dirk Soffker. Comparison of different model-free control methods concerning real-time benchmark. *Journal of Dynamic Systems, Measurement, and Control*, 140(12), August 2018.

[69] Anh-Tu Nguyen, Tadanari Taniguchi, Luka Eciolaza, Victor Campos, Reinaldo Palhares, and Michio Sugeno. Fuzzy control systems: Past, present and future. *IEEE Computational Intelligence Magazine*, 14(1):56–68, February 2019.

[70] Robert John and Simon Coupland. Type-2 fuzzy logic: A historical view. *IEEE Computational Intelligence Magazine*, 2(1):57–62, February 2007.

[71] Jyh-Shing R. Jang and Chuen-Tsai Sun. Neuro-fuzzy modeling and control. *Proceedings of the IEEE*, 83(3):378–406, 1995.

[72] Hamid R. Berenji and Pratap Khedkar. Learning and tuning fuzzy logic controllers through reinforcements. *IEEE Transactions on Neural Networks*, 3(5):724–740, September 1992.

[73] Leandro dos Santos Coelho, Marcelo Witchtoff Pessoa, Antonio Augusto Rodrigues Coelho, and Rodrigo Rodrigues Sumar. Model-free adaptive control design using evolutionary-neural compensator. *Expert Systems with Applications*, 37:499–508, 2010.

[74] Abhijit Das, Frank L. Lewis, and Kamesh Subbarao. Backstepping approach for controlling a quadrotor using lagrange form dynamics. *Journal of Intelligent and Robotic Systems*, 56(1-2):127–151, September 2009.

[75] Naser El-Sheimy, Kai-Wei Chiang, and Aboelmagd Noureldin. The utilization of artificial neural networks for multisensor system integration in navigation and positioning instruments. *IEEE Transactions on Instrumentation and Measurement*, 55(5):1606–1615, Oct 2006.

[76] Andrew G. Barto, Richard S. Sutton, and Charles W. Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-13(5):834–846, September 1983.

[77] Matthew Cook. It takes two neurons to ride a bicycle. In *Proceedings of the 2004 Conference on Neural Information Processing Systems*, 2004.

[78] Kai Arulkumaran, Marc P. Deisenroth, Miles Brundage, and Anil A. Bharath. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, 34(6):26–38, November 2017.

[79] Lucian Busoniu, Robert Babuska, and Bart De Schutter. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 38(2):156–172, March 2008.

[80] Murad Abu-Khalaf, Frank L. Lewis, and Jie Huang. Neurodynamic programming and zero-sum games for constrained control systems. *IEEE Transactions on Neural Networks*, 19(7):1243–1252, July 2008.

[81] Mohammed Abouheaf and Wail Gueaieb. Model-free adaptive control approach using integral reinforcement learning. In *2019 IEEE International Symposium on Robotic and Sensors Environments (ROSE)*, pages 1–7, June 2019.

[82] Richard E. Bellman. *Dynamic Programming*. Princeton University Press, 1957.

[83] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Massachusetts, second edition, 1998.

[84] Warren B. Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. Wiley, 2nd edition, 2011.

[85] Paul Werbos. *Beyond Regression: New Tools for Prediction and Analysis in the Behavior Sciences*. PhD thesis, Harvard University, 1974.

[86] Paul J. Werbos. Neural networks for control and system identification. In *28th Conference on Decision and Control*, pages 260–265, December 1989.

[87] Paul J. Werbos. A menu of designs for reinforcement learning over time. In *Neural Networks for Control*, pages 67–95. MIT Press, Cambridge, MA, USA, 1990.

[88] Paul J. Werbos. Neurocontrol and supervised learning: An overview and evaluation. In *Handbook of Intelligent Control: Neural, Fuzzy, and Adaptive Approaches*, pages 65–89. Van Nostrand Reinhold, June 1992.

[89] Richard S. Sutton. Learning to predict by the methods of temporal difference. *Machine Learning*, 3:9–44, 1988.

[90] Christopher J. C. H. Watkins and Peter Dayan. Q-learning. *Machine Learning*, 8(3):279–292, May 1992.

[91] Dimitri P. Bertsekas and John N. Tsitsiklis. Neuro-dynamic programming: An overview. In *Proceedings of the IEEE Conference on Decision and Control*, pages 560–564, Dec 1995.

[92] Dimitri P. Bertsekas and John N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, Massachusetts, 1st edition, 1996.

[93] Kenji Doya. Temporal difference learning in continuous time and space. In *Proceedings of the 1995 Conference: Advances in Neural Information Processing Systems*, pages 1073–1079, Cambridge, MA, 1995.

[94] Andrew G. Barto, Steven J. Bradtke, and Satinder P. Singh. Learning to act using real-time dynamic programming. *Artificial Intelligence*, 72:81–138, 1995.

[95] Frank L. Lewis, Kai Liu, and Aydin Yesildirek. Neural net robot controller with guaranteed tracking performance. *IEEE Transactions on Neural Networks*, 6(3):703–715, May 1995.

[96] Frank L. Lewis, Aydin Yesildirek, and Kai Liu. Multilayer neural-net robot controller with guaranteed tracking performance. *IEEE Transactions on Neural Networks*, 7(2):388–399, March 1996.

[97] Said G. Khan, Guido Herrmann, Frank L. Lewis, Tony Pipe, and Chris Melhuish. Reinforcement learning and optimal adaptive control: An overview and implementation examples. *Annual Reviews in Control*, 36(1):42–59, April 2012.

[98] Tomas Landelius and Hans Knutsson. Greedy adaptive critics for LQR problems: Convergence proofs. Linköping University, January 1996.

[99] Danil V. Prokhorov and Donald C. Wunsch. Adaptive critic designs. *IEEE Transactions on Neural Networks*, 8(5):997–1007, September 1997.

[100] Bernard Widrow, Narendra K. Gupta, and Sidhartha Maitra. Punish/reward: Learning with a critic in adaptive threshold systems. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-3(5):455–465, 1973.

[101] Frank L. Lewis and Draguna Vrabie. Reinforcement learning and adaptive dynamic programming for feedback control. *IEEE Circuits and Systems Magazine*, 9(3):32–50, 2009.

[102] Frank L. Lewis and Derong Liu. *Reinforcement Learning and Approximate Dynamic Programming for Feedback Control*. John Wiley & Sons, Hoboken, NJ, 2013.

[103] Frank L. Lewis, Draguna Vrabie, and Kyriakos G. Vamvoudakis. Reinforcement learning and feedback control: Using natural decision methods to design optimal adaptive controllers. *IEEE Control Systems Magazine*, 32(6):76–105, December 2012.

[104] Sivasubramanya N. Balakrishnan, Jie Ding, and Frank L. Lewis. Issues on stability of ADP feedback controllers for dynamical systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 38(4):913–917, August 2008.

[105] Asma Al-Tamimi, Frank L. Lewis, and Murad Abu-Khalaf. Discrete-time nonlinear HJB solution using approximate dynamic programming: Convergence proof. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 38(4):943–949, 2008.

[106] Asma Al-Tamimi, Murad Abu-Khalaf, and Frank L. Lewis. Adaptive critic designs for discrete-time zero-sum games with application to H_∞ control. *IEEE Transactions on Systems, Man, and Cybernetics*, 37(1):240–247, February 2007.

[107] Asma Al-Tamimi, Frank L. Lewis, and Murad Abu-Khalaf. Model-free Q-learning designs for linear discrete-time zero-sum games with application to H-infinity control. *Automatica*, 43(3):473–481, March 2007.

[108] Draguna Vrabie and Frank L. Lewis. Neural network approach to continuous-time direct adaptive optimal control for partially unknown nonlinear systems. *Neural Networks*, 22(3):237–246, April 2009.

[109] Frank L. Lewis and Kyriakos G. Vamvoudakis. Reinforcement learning for partially observable dynamic processes: Adaptive dynamic programming using measured output data. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 41(1):14–25, February 2011.

[110] Hamidreza Modares, Frank L. Lewis, and Mohammad-Bagher Naghibi-Sistani. Integral reinforcement learning and experience replay for adaptive optimal control of partially-unknown constrained-input continuous-time systems. *Automatica*, 50(1):193–202, January 2014.

[111] Hamidreza Modares and Frank L. Lewis. Linear quadratic tracking control of partially-unknown continuous-time systems using reinforcement learning. *IEEE Transactions on Automatic Control*, 59(11):3051–3056, November 2014.

[112] Yu Jiang and Zhong-Ping Jiang. Computational adaptive optimal control for continuous-time linear systems with completely unknown dynamics. *Automatica*, 48(10):2699–2704, October 2012.

[113] Jae Young Lee, Jin B. Park, and Yoon H. Choi. Integral Q-learning and explorized policy iteration for adaptive optimal control of continuous-time linear systems. *Automatica*, 48(11):2850–2859, November 2012.

[114] Bahare Kiumarsi, Frank L. Lewis, Hamidreza Modares, Ali Karimpour, and Mohammad-Bagher Naghibi-Sistani. Reinforcement Q-learning for optimal tracking control of linear discrete-time systems with unknown dynamics. *Automatica*, 50(4):1167–1175, April 2014.

[115] Hamidreza Modares, Frank L. Lewis, and Zhong-Ping Jiang. H_∞ tracking control of completely unknown continuous-time systems via off-policy reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems*, 26(10):2550–2562, October 2015.

[116] Yongliang Yang, Yixin Yin, Wei He, Kyriakos G. Vamvoudakis, Hamidreza Modares, and Donald C. Wunsch. Safety-aware reinforcement learning framework with an actor-critic-barrier structure. In *2019 American Control Conference (ACC)*, pages 2352–2358, 2019.

[117] Navid Moshtaghi Yazdani, Reihaneh Kardehi Moghaddam, Bahare Kiumarsi, and Hamidreza Modares. A safety-certified policy iteration algorithm for control of constrained nonlinear systems. *IEEE Control Systems Letters*, 4(3):686–691, 2020.

[118] Muddasar Naeem, Syed T. H. Rizvi, and Antonio Coronato. A gentle introduction to reinforcement learning and its application in different fields. *IEEE Access*, 8:209320–209344, 2020.

[119] Jennie Si and Yu-Tsung Wang. Online learning control by association and reinforcement. *IEEE Transactions on Neural Networks*, 12(2):264–276, March 2001.

[120] Draguna Vrabie, Octavian Pastravanu, Murad Abu-Khalaf, and Frank L. Lewis. Adaptive optimal control for continuous-time linear systems based on policy iteration. *Automatica*, 45(2):477–484, February 2009.

[121] Kyriakos Vamvoudakis, Draguna Vrabie, and Frank L. Lewis. Online policy iteration based algorithms to solve the continuous-time infinite horizon optimal control problem. In *2009 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning*, pages 36–41, March 2009. ISSN: 2325-1867.

[122] Kyriakos G. Vamvoudakis and Frank L. Lewis. Online actor-critic algorithm to solve the continuous-time infinite horizon optimal control problem. *Automatica*, 46(5):878–888, May 2010.

[123] Shangke Lyu and Chien C. Cheah. Data-driven learning for robot control with unknown Jacobian. *Automatica*, 120:109120, October 2020.

[124] Shai A. Arogeti and Frank L. Lewis. Static Output-Feedback H_∞ control design procedures for continuous-time systems with different levels of model knowledge. *IEEE Transactions on Cybernetics*, 53(3):1432–1446, March 2023.

[125] Majid Mazouchi, Subramanya Nageshrao, and Hamidreza Modares. Conflict-aware safe reinforcement learning: A meta-cognitive learning framework. *IEEE/CAA Journal of Automatica Sinica*, 9(3):466–481, 2022.

[126] Shuzheng Qu, Mohammed Abouheaf, Wail Gueaieb, and Davide Spinello. A policy iteration approach for flock motion control. In *2021 IEEE International Symposium on Robotic and Sensors Environments (ROSE)*, pages 1–7, 2021.

[127] Adekunle A. Adepegba, Md Suruz Miah, and Davide Spinello. Multi-agent area coverage control using reinforcement learning techniques. In *Proceedings of the 29th International Florida Artificial Intelligence Research Society Conference*, 2016.

[128] Farzan Soleymani, Md Suruz Miah, and Davide Spinello. Optimal non-autonomous area coverage control with adaptive reinforcement learning. *Engineering Applications of Artificial Intelligence*, 122:106068, 2023.

[129] Kyriakos G. Vamvoudakis, Frank L. Lewis, and Greg R. Hudas. Multi-agent differential graphical games: Online adaptive learning solution for synchronization with optimality. *Automatica*, 48(8):1598–1611, August 2012.

[130] Mohammed I. Abouheaf, Frank L. Lewis, Kyriakos G. Vamvoudakis, Sofie Haesaert, and Robert Babuska. Multi-agent discrete-time graphical games and reinforcement learning solutions. *Automatica*, 50(12):3038–3053, December 2014.

[131] Mohammed I. Abouheaf and Frank L. Lewis. Multi-agent differential graphical games: Nash online adaptive learning solutions. In *52nd IEEE Conference on Decision and Control*, pages 5803–5809, Dec 2013.

[132] Mohammed Abouheaf and Frank Lewis. Approximate dynamic programming solutions of multi-agent graphical games using actor-critic network structures. In *International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, August 2013.

[133] Mohammed Abouheaf and Frank L. Lewis. Dynamic graphical games: Online adaptive learning solutions using approximate dynamic programming. In *Frontiers of Intelligent Control and Information Processing*, pages 1–48. World Scientific, 2014.

[134] Mohammed I. Abouheaf, Frank L. Lewis, Kyriakos Vamvoudakis, Sofie Haesaert, and Robert Babuska. Multi-agent discrete-time graphical games and reinforcement learning solutions. *Automatica*, 50(12):3038–3053, 2014.

[135] Mohammed I. Abouheaf, Frank L. Lewis, and Magdi. S. Mahmoud. Differential graphical games: Policy iteration solutions and coupled riccati formulation. In *2014 European Control Conference (ECC)*, pages 1594–1599, June 2014.

[136] Mohammed I. Abouheaf, Frank L. Lewis, and Magdi. S. Mahmoud. Model-free adaptive learning solutions for discrete-time dynamic graphical games. In *53rd IEEE Conference on Decision and Control*, pages 3578–3583, Dec 2014.

[137] Mohammed I. Abouheaf, Frank L. Lewis, Magdi S. Mahmoud, and Dariusz G. Mikulski. Discrete-time dynamic graphical games: Model-free reinforcement learning solution. *Control Theory and Technology*, 13(1):55–69, February 2015.

[138] Mohammed Abouheaf and Wail Gueaieb. Multi-agent reinforcement learning approach based on reduced value function approximations. In *IEEE International Symposium on Robotics and Intelligent Sensors (IRIS)*, pages 111–116, October 2017.

[139] Mohammed I. Abouheaf and Magdi S. Mahmoud. Policy iteration and coupled riccati solutions for dynamic graphical games. *International Journal of Digital Signals and Smart Systems*, 1(2):143–162, 2017.

[140] Andrew G. Barto, Richard S. Sutton, and Charles W. Anderson. Looking back on the actor-critic architecture. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 51(1), January 2021.

[141] Francis M. Rogallo. Nasa Research on Flexible Wings. *Annals of the New York Academy of Sciences*, 154(2):953–961, 1968.

[142] E. A. Kilkenny. *An experimental study of the longitudinal aerodynamic and static stability characteristics of hang gliders*. PhD thesis, College of Aeronautics, Cranfield Institute of Technology, September 1986.

[143] Ilan Kroo and Li-Shing Chang. Analytical and scale model research aimed at improved hangglider design. In *NASA Langley Research Center of Science and Technology of Low Speed and Motorless Flight*, June 1979.

[144] Ilan Kroo. Aerodynamics, aeroelasticity, and stability of hang gliders: Experimental results. Technical Report NASA-TM-81269, NASA Ames Research Center, Moffett Field, CA, April 1981.

[145] Michael V. Cook. A summary report on an experimental investigation into methods for quantifying hang glider airworthiness parameters. Technical Report 8705, College of Aeronautics, Cranfield Institute of Technology, May 1987.

[146] E. A Kilkenny. An evaluation of a mobile aerodynamic test facility for hang glider wings. Technical Report 8330, College of Aeronautics, Cranfield Institute of Technology, 1983.

[147] E. A. Kilkenny. Full scale wind tunnel tests on hang glider pilots. Technical Report 8416, College of Aeronautics, Cranfield Institute of Technology, 1984.

[148] Guy B. Gratton and Simon Newman. The 'tumble' departure mode in weightshift-controlled microlight aircraft. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 217(3):149–166, 2003.

[149] Guido De Matteis. Dynamics of hang gliders. *Journal of Guidance Control and Dynamics*, 14(6):1145–1152, 1991.

[150] Guido De Matteis. Response of hang gliders to control. *The Aeronautical Journal*, 94(938):289–294, 1990.

[151] Guido De Matteis. Hang-glider response to atmospheric inputs. *Journal of Guidance, Control, and Dynamics*, 15(4):1048–1050, July 1992.

[152] Wael Khaddage. A wind-tunnel investigation of an ultra-light wing and ultra-light aircraft. Master's thesis, Carleton University, 2017.

[153] Michael V. Cook. The theory of the longitudinal static stability of the hang-glider. *The Aeronautical Journal*, 98(978):292–304, 1994.

[154] R. Rollins. Study of experimental data to assess the longitudinal stability and control of the hang glider. M.sc. thesis, College of Aeronautics, Cranfield Institute of Technology, 2000. Digitization funded by Nathaniel Mailhot, available on request.

[155] M. Spottiswoode. A theoretical study of the lateral-directional dynamics, stability and control of the hang glider. M.sc. thesis, College of Aeronautics, Cranfield Institute of Technology, 2001. Digitization funded by Nathaniel Mailhot, available on request.

[156] Michael V. Cook. *Flight Dynamics Principles: A Linear Systems Approach to Aircraft Stability and Control*. Aerospace Engineering. Butterworth-Heinemann, 3rd edition, 2013.

[157] Yoshimasa Ochi. Modeling of the longitudinal dynamics of a hang glider. In *AIAA Modeling and Simulation Technologies Conference*, pages 1591–1608, 2015.

[158] Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, and Giuseppe Oriolo. *Robotics*. Advanced Textbooks in Control and Signal Processing. Springer, London, 2009.

[159] Jianqing Peng, Wenfu Xu, Fengxu Wang, Yu Han, and Bin Liang. A hybrid hand-eye calibration method for multilink cable-driven hyper-redundant manipulators. *IEEE Transactions on Instrumentation and Measurement*, 70:1–13, 2021.

[160] Minh Hung Vu and Uhn Joo Na. A new 6-DOF haptic device for teleoperation of 6-DOF serial robots. *IEEE Transactions on Instrumentation and Measurement*, 60(11):3510–3523, November 2011.

[161] Haider A. F. Almurib, Haidar F. Al-Qrimli, and Nandha Kumar. A review of application industrial robotic design. In *2011 Ninth International Conference on ICT and Knowledge Engineering*, pages 105–112, 2012.

[162] Naser Mostashiri, Jaspreet S. Dhupia, Alexander W. Verl, and Weiliang Xu. A review of research aspects of redundantly actuated parallel robotsw for enabling further applications. *IEEE/ASME Transactions on Mechatronics*, 23(3):1259–1269, 2018.

[163] Mahmoud Zarebidoki, Jaspreet S. Dhupia, and Weiliang Xu. A review of cable-driven parallel robots: Typical configurations, analysis techniques, and control methods. *IEEE Robotics & Automation Magazine*, 29(3):89–106, 2022.

[164] Andreas Pott, Hendrick Mütherich, Werner Kraus, Valentine Schmidt, Philipp Miermeister, and Alexander Verl. IPAnema: A family of cable-driven parallel robots for industrial applications. In *Cable-Driven Parallel Robots*, Mechanisms and Machine Science, pages 119–134. Berlin, Heidelberg, 2013.

[165] Francois Pierrot, Vincent Nabat, Olivier Company, Sebastien Krut, and Philippe Poignet. Optimal design of a 4-DOF parallel manipulator: From academia to industry. *IEEE Transactions on Robotics*, 25(2):213–224, 2009.

[166] Richard Verhoeven. *Analysis of the Workspace of Tendon-based Stewart Platforms*. PhD thesis, Universität Duisburg-Essen, Duisburg, Germany, July 2004.

[167] Marco Carricato and Jean-Pierre Merlet. Stability analysis of underconstrained cable-driven parallel robots. *IEEE Transactions on Robotics*, 29(1):288–296, February 2013.

[168] Y.K. Yiu, H. Cheng, Z.H. Xiong, Guanfeng L. Liu, and Z.X. Li. On the dynamics of parallel manipulators. In *Proceedings IEEE International Conference on Robotics and Automation*, volume 4, pages 3766–3771 vol.4, 2001.

[169] Rodney G. Roberts, Todd Graham, and Thomas Lippitt. On the inverse kinematics, statics, and fault tolerance of cable-suspended robots. *Journal of Robotic Systems*, 15(10):581–597, 1998.

[170] Abdullah B. Alp and Sunil K. Agrawal. Cable suspended robots: Design, planning and control. In *Proceedings 2002 IEEE International Conference on Robotics and Automation*, volume 4, pages 4275–4280, May 2002.

[171] Ali Afshari and Ali Meghdari. New Jacobian matrix and equations of motion for a 6 D.O.F cable-driven robot. *International Journal of Advanced Robotic Systems*, 4(1):8, March 2007.

[172] Sen Qian, Bin Zi, Wei-Wei Shang, and Qing-Song Xu. A review on cable-driven parallel robots. *Chinese journal of mechanical engineering*, 31(1):29–39, 2018.

[173] Imme Ebert-Uphoff and Philip A. Voglewede. On the connections between cable-driven robots, parallel manipulators and grasping. In *Proceedings IEEE International Conference on Robotics and Automation*, volume 5, pages 4521–4526, 2004.

[174] Muneaki Miyasaka, Mohammad Haghighipanah, Yangming Li, and Blake Hannaford. Hysteresis model of longitudinally loaded cable for cable driven robots and identification of the parameters. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4051–4057, May 2016.

[175] Thibaut Paty, Nicolas Binaud, Stephane Caro, and Stephane Segonds. Cable-driven parallel robot modelling considering pulley kinematics and cable elasticity. *Mechanism and machine theory*, 159:104263, 2021.

[176] Patricia Ben-Horin and Federico Thomas. A nonholonomic 3-DOF parallel robot. In *Advances in Robot Kinematics: Analysis and Design*, pages 111–118. Springer Netherlands, Dordrecht, 2008.

[177] Hamed Khakpour, Lionel Birglen, and Souheil-Antoine Tahan. Synthesis of differentially driven planar cable parallel manipulators. *IEEE Transactions on Robotics*, 30(3):619–630, 2014.

[178] Luca Barbazza, Fabio Oscari, Simone Minto, and Giulio Rosati. Trajectory planning of a suspended cable driven parallel robot with reconfigurable end effector. *Robotics and Computer-Integrated Manufacturing*, 48:1–11, December 2017.

[179] Ryan J. Caverly and James R. Forbes. Flexible cable-driven parallel manipulator control: Maintaining positive cable tensions. *IEEE Transactions on Control Systems Technology*, 26(5):1874–1883, September 2018.

[180] Weiwei Shang, Bingyuan Zhang, Bin Zhang, Fei Zhang, and Shuang Cong. Synchronization control in the cable space for cable-driven parallel robots. *IEEE Transactions on Industrial Electronics*, 66(6):4544–4554, June 2019.

[181] Hamid R. Fahham and Mehrdad Farid. Minimum-time trajectory planning of spatial cable-suspended robots along a specified path considering both tension and velocity constraints. *Engineering Optimization*, 42(4):387–402, April 2010.

[182] Xiaoling Jiang and Clement Gosselin. Dynamic point-to-point trajectory planning of a three-DOF cable-suspended parallel robot. *IEEE Transactions on Robotics*, 32(6):1550–1557, December 2016.

[183] Nan Zhang and Weiwei Shang. Dynamic trajectory planning of a 3-DOF under-constrained cable-driven parallel robot. *Mechanism and Machine Theory*, 98:21–35, April 2016.

[184] Bin Zhang, Weiwei Shang, Shuang Cong, and Zhijun Li. Coordinated dynamic control in the task space for redundantly actuated cable-driven parallel robots. *IEEE/ASME Transactions on Mechatronics*, 26(5):2396–2407, October 2021.

[185] Deng Lin and Giovanni Mottola. Dynamic launch trajectory planning of a cable-suspended translational parallel robot using point-to-point motions. *Machines*, 11(2):224, February 2023.

[186] Ali Ghasemi, Mohammad Eghtesad, and Mehrdad Farid. Neural network solution for forward kinematics problem of cable robots. *Journal of Intelligent & Robotic Systems*, 60(2):201–215, November 2010.

[187] Moharam. H. Korayem, Hami Tourajizadeh, Ashkan Zehfroosh, and Amin H. Korayem. Optimal path planning of a cable-suspended robot with moving boundary using optimal feedback linearization approach. *Nonlinear Dynamics*, 78(2):1515–1543, October 2014.

[188] Udwadia-Kalaba approach for parallel manipulator dynamics. 135(6):061003.

[189] Rongrong Yu, Han Zhao, Shengchao Zhen, and Kang Huang. A novel approach for 2-degrees of freedom redundant parallel manipulator dynamics. *Advances in Mechanical Engineering*, 9(6), June 2017.

[190] Chenming Li, Han Zhao, Shengchao Zhen, Kang Huang, Hao Sun, Ke Shao, and Bin Deng. Trajectory tracking control of parallel manipulator based on Udwadia-Kalaba approach. *Mathematical Problems in Engineering*, 2017:e8975743, December 2017.

[191] Zicheng Zhu, Shengchao Zhen, Zhihao Chen, and Hao Sun. Udwadia-Kalaba control approach for 3-DOF planar parallel manipulator. In *2019 1st International Conference on Industrial Artificial Intelligence (IAI)*, pages 1–6, July 2019.

[192] Firdaus E. Udwadia and Robert E. Kalaba. *Analytical Dynamics: A New Approach*. Cambridge University Press, Cambridge, 1996.

[193] Caner Sancak, Mehmet Itik, and Thang T. Nguyen. Position control of a fully constrained planar cable-driven parallel robot with unknown or partially known dynamics. *IEEE/ASME Transactions on Mechatronics*, pages 1–11, 2022.

[194] Hamed Jabbari Asl and Jungwon Yoon. Robust trajectory tracking control of cable-driven parallel robots. *Nonlinear Dynamics*, 89(4):2769–2784, September 2017.

[195] Hamed Jabbari Asl and Farrokh Janabi-Sharifi. Adaptive neural network control of cable-driven parallel robots with input saturation. *Engineering Applications of Artificial Intelligence*, 65:252–260, October 2017.

[196] George K. I. Mann and Brian W. Surgenor. Model-free intelligent control of a 6-DOF Stewart-Gough based parallel manipulator. In *Proceedings of the International Conference on Control Applications*, volume 1, pages 495–500 vol.1, 2002.

[197] Hao Xiong, Lin Zhang, and Xiumin Diao. A learning-based control framework for cable-driven parallel robots with unknown Jacobians. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 234(9):1024–1036, 2020.

[198] Mohammad O. Tokhi and Abul K. M. Azad, editors. *Flexible Robot Manipulators: Modelling, Simulation and Control*. IET Digital Library, January 2008.

[199] John R. Clymer. *Simulation-Based Engineering of Complex Systems*. Wiley, 2nd edition, 2009.

[200] Olivier Parodi, Lionel Lapierre, and Bruno Jouvencel. Hardware-in-the-loop simulators for multi-vehicles scenarios: Survey on existing solutions and proposal of a new architecture. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 225–230, October 2009.

[201] Serena Ivaldi, Jan Peters, Vincent Padois, and Francesco Nori. Tools for simulating humanoid robot dynamics: A survey based on user feedback. In *2014 IEEE-RAS International Conference on Humanoid Robots*, pages 842–849, November 2014.

[202] Wenshuai Zhao, Jorge Peña Queralta, and Tomi Westerlund. Sim-to-Real transfer in deep reinforcement learning for robotics: A survey. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 737–744, December 2020.

[203] Hesham Alghodhaifi and Sridhar Lakshmanan. Autonomous vehicle evaluation: A comprehensive survey on modeling and simulation approaches. *IEEE Access*, 9:151531–151566, 2021.

[204] Aaron Staranowicz and Gian L. Mariottini. A survey and comparison of commercial and open-source robotic simulator software. In *Proceedings of the 4th International Conference on PErvasive Technologies Related to Assistive Environments (PETRA)*, pages 1–8, New York, NY, USA, May 2011.

[205] Daniel Cook, Andrew Vardy, and Ron Lewis. A survey of AUV and robot simulators for multi-vehicle operations. In *2014 IEEE/OES Autonomous Underwater Vehicles (AUV)*, pages 1–8, October 2014.

[206] Farzan M. Noori, David Portugal, Rui P. Rocha, and Micael S. Couceiro. On 3D simulators for multi-robot systems in ROS: MORSE or Gazebo? In *2017 IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*, pages 19–24, October 2017.

[207] Nathan P. Koenig and Andrew Howard. Design and use paradigms for Gazebo, an open-source multi-robot simulator. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 3, pages 2149–2154 vol.3, September 2004.

[208] Jeff Craighead, Robin Murphy, Jenny Burke, and Brian Goldiez. A survey of commercial & open source unmanned vehicle simulators. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 852–857, April 2007.

[209] Jason M. O'Kane. *A Gentle Introduction to ROS*. CreateSpace Independent Publishing Platform, October 2013. `http://www.cse.sc.edu/~jokane/agitr/`.

[210] Anis Koubaa, editor. *Robot Operating System (ROS)*, volume 778 of *Studies in Computational Intelligence*. Springer International Publishing, 2019.

[211] Unmanned ground vehicle modelling in Gazebo/ROS-Based Environments, volume = 7, copyright = http://creativecommons.org/licenses/by/3.0/, issn = 2075-1702, url = https://www.mdpi.com/2075-1702/7/2/42, doi = 10.3390/machines7020042, language = en, number = 2, urldate = 2023-03-13, journal = Machines, author = Rivera, Zandra B. and De Simone, Marco C. and Guida, Domenico, month = jun, year = 2019, keywords = gazebo, Matlab, multibody dynamics, robotics, wheeled mobile robot, pages = 42, file = Full Text PDF:/Users/nathanielmailhot/Zotero/storage/X2ME64RL/Rivera et al. - 2019 - Unmanned Ground Vehicle Modelling in GazeboROS-Ba.pdf:application/pdf,.

[212] Eric Rohmer, Surya P. N. Singh, and Marc Freese. V-REP: A versatile and scalable robot simulation framework. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1321–1326, November 2013.

[213] Olivier Michel. Cyberbotics Ltd. Webots: Professional mobile robot simulation. *International Journal of Advanced Robotic Systems*, 1(1):5, March 2004.

[214] Marwah M. Almasri, Abrar M. Alajlan, and Khaled M. Elleithy. Trajectory planning and collision avoidance algorithm for mobile robotics system. *IEEE Sensors Journal*, 16(12):5021–5028, June 2016.

[215] Behzad Boroujerdian, Hasan Genc, Srivatsan Krishnan, Wenzhi Cui, Aleksandra Faust, and Vijay Reddi. MAVBench: Micro aerial vehicle benchmarking. In *2018 51st Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pages 894–907, October 2018.

[216] Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. AirSim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and Service Robotics*, Springer Proceedings in Advanced Robotics, pages 621–635, 2018.

[217] Jonathan Platt and Kenneth Ricks. Comparative analysis of ROS-Unity3D and ROS-Gazebo for mobile ground robot simulation. *Journal of Intelligent & Robotic Systems*, 106(4):80, December 2022.

[218] Chi Zhang, Yuehu Liu, Danchen Zhao, and Yuanqi Su. RoadView: A traffic scene simulator for autonomous vehicle simulation testing. In *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 1160–1165, October 2014.

[219] Wei Qian, Zeyang Xia, Jing Xiong, Yangzhou Gan, Yangchao Guo, Shaokui Weng, Hao Deng, Ying Hu, and Jianwei Zhang. Manipulation task simulation using ROS

and Gazebo. In *2014 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 2594–2598, December 2014.

[220] Carlos E. Agüero, Nate Koenig, Ian Chen, Hugo Boyer, Steven Peters, John Hsu, Brian Gerkey, Steffi Paepcke, Jose L. Rivero, Justin Manzo, Eric Krotkov, and Gill Pratt. Inside the virtual robotics challenge: Simulating real-time robotic disaster response. *IEEE Transactions on Automation Science and Engineering*, 12(2):494–506, April 2015.

[221] Lei Tai and Ming Liu. A robot exploration strategy based on Q-learning network. In *2016 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, pages 57–62, June 2016.

[222] Wei Li and Rong Xiong. Dynamical obstacle avoidance of task-constrained mobile manipulation using model predictive control. *IEEE Access*, 7:88301–88311, 2019.

[223] Musa M. M. Manhães, Sebastian A. Scherer, Martin Voss, Luiz R. Douat, and Thomas Rauschenbach. UUV simulator: A Gazebo-based package for underwater intervention and multi-robot simulation. pages 1–8, September 2016.

[224] Nursultan Imanberdiyev, Changhong Fu, Erdal Kayacan, and I-Ming Chen. Autonomous navigation of UAV by using real-time model-based reinforcement learning. In *14th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, pages 1–6, November 2016.

[225] Sohan Suvarna, Dibyayan Sengupta, Pavankumar Koratikere, and Rajkumar S. Pant. Simulation of autonomous airship on ROS-Gazebo framework. In *2019 Fifth Indian Control Conference (ICC)*, pages 237–241, January 2019.

[226] Antun Ivanovic, Marsela Polic, Jelena Tabak, and Matko Orsag. Render-in-the-loop aerial robotics simulator: Case study on yield estimation in indoor agriculture. In *2022 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 787–793, June 2022.

[227] Louis. P. M. Nelissen. Development of Gazebo-based high-fidelity simulation environment for morphing-wing UAVs. University of Twente, Enschede, Netherlands, `https://essay.utwente.nl/89215/`, December 2021.

[228] Matt Schmittle, Anna Lukina, Lukas Vacek, Jnaneshwar Das, Christopher P. Buskirk, Stephen Rees, Janos Sztipanovits, Radu Grosu, and Vijay Kumar. OpenUAV: A UAV testbed for the CPS and robotics community. In *ACM/IEEE 9th*

International Conference on Cyber-Physical Systems (ICCPS), pages 130–139, April 2018.

[229] Rhys Mainwaring and ArduPilot Dev Team. ArduPilot Gazebo Plugin. `https://github.com/ArduPilot/ardupilot_gazebo`, May 2024. Accessed: May 30, 2024.

[230] Timothy H. Chung, Michael R. Clement, Michael A. Day, Kevin D. Jones, Duane Davis, and Marianna Jones. Live-fly, large-scale field experimentation for large numbers of fixed-wing UAVs. *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1255–1262, May 2016.

[231] Duane T. Davis, Timothy H. Chung, Michael R. Clement, and Michael A. Day. Multi-swarm infrastructure for swarm versus swarm experimentation. In *Distributed Autonomous Robotic Systems: The 13th International Symposium*, Springer Proceedings in Advanced Robotics, pages 649–663. 2018.

[232] Emlid Limited. Emlid open-source documentation. `https://github.com/emlid/emlid-docs`. Accessed: March 15, 2023.

[233] Raspberry Pi Foundation. Raspberry pi documentation. `https://github.com/raspberrypi/documentation`. Accessed: March 15, 2023.

[234] ArduPilot Development Team and Community. ArduPilot - open source autopilot. `http://ardupilot.org/`. Accessed: March 15, 2023.

[235] BYU MAGICC Lab. ROS Flight. `https://rosflight.org/`. Accessed: March 15, 2023.

[236] TDK Corp. InvenSense. MPU-9250 sensor documentation. `https//www.invensense.com/products/motion-tracking-9-axis/mpu-9250/`. Accessed: December 1, 2022.

[237] STMicroelectronics N.V. LSM9DS1 sensor documentation. `https://www.st.com/en/mems-and-sensors/lsm9ds1.html`. Accessed: January 15th.

[238] Robotis Co. Ltd. Dynamixel SDK. `http://emanual.robotis.com/docs/en/software/dynamixel`. Accessed: March 15, 2023.

[239] Wills Wing. Falcon 3. `https://www.willswing.com/hang-gliders/falcon-3/`. Accessed: March 15, 2023.

[240] XFLR5 Development Team. XFLR5: Analysis tool for airfoils, wings, and planes. http://www.xflr5.tech/xflr5.htm, 2022. Accessed: March 15, 2023.

[241] Meiya Group. Cosmos aerospace overview. http://meiyagroup.com/cosmos-aerospace.html. Accessed: July 19, 2023.

[242] La Mouette. Ultralight wings. https://lamouette.com/en/the-ultralight-wing-ranges/. Accessed: July 19, 2023.

[243] Samuel Tabor, Iain Guilliard, and Andrey Kolobov. Ardusoar: An open-source thermalling controller for resource-constrained autopilots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 6255–6262, October 2018.

[244] Hugo Andrade de Oliveira and Paulo Fernando Ferreira Rosa. Genetic neuro-fuzzy approach for unmanned fixed wing attitude control. In *International Conference on Military Technologies*, pages 485–492, May 2017.

[245] Transport Canada. Recreational aircraft airworthiness. https://tc.canada.ca/en/aviation/aircraft-airworthiness/recreational-aircraft-airworthiness, 2019. Accessed: April 10th, 2024.

[246] Ultralight Pilots Association of Canada. Light aircraft manufacturers of Canada (LAMAC). https://upac.ca/light-aircraft-manufacturers-of-canada-lamac/, 2024. Accessed: April 10th, 2024.

[247] Federal Aviation Administration. eCFR - Code of federal regulations, Title 14, Part 103: Ultralight vehicles. https://www.ecfr.gov/current/title-14/chapter-I/subchapter-F/part-103?toc=1, 2023. Accessed: April 10th, 2024.

[248] Federal Aviation Administration. eCFR - Code of federal regulations, Title 14, Part 21.190: Issue of a special airworthiness certificate for a light-sport category aircraft. https://www.ecfr.gov/current/title-14/chapter-I/subchapter-C/part-21/subpart-H/section-21.190, 2023. Accessed: April 10th, 2024.

[249] Agencia Federal de Aviación Civil. Reglamentos. https://www.gob.mx/afac/acciones-y-programas/reglamentos-252211, 2023. Accessed: April 10th, 2024.

[250] Google LLC. Google Earth. https://www.google.com/earth/, 2021.

[251] Joshua L. Proctor, Steven L. Brunton, and J. Nathan Kutz. Dynamic mode decomposition with control. *SIAM J. Appl. Dyn. Syst.*, 15(1):142–161, Jan 2016.

[252] Diya Sashidhar and J. Nathan Kutz. Bagging, optimized dynamic mode decomposition (BOP-DMD) for robust, stable forecasting with spatial and temporal uncertainty-quantification. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 380(2229):20210199, August 2022.

[253] Meghana Velegar, Christoph Keller, and J. Nathan Kutz. Optimized dynamic mode decomposition for reconstruction and forecasting of atmospheric chemistry data, April 2024.

[254] Travis Askham and J. Nathan Kutz. Variable projection methods for an optimized dynamic mode decomposition. *SIAM Journal on Applied Dynamical Systems*, 17(1):380–416, 2018.

[255] Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences of the United States of America*, 113(15):3932–3937, Apr 2016.

[256] Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Sparse identification of nonlinear dynamics with control (SINDYc). *IFAC-PapersOnLine: 10th IFAC Symposium on Nonlinear Control Systems (NOLCOS)*, 49(18):710–715, 2016.

[257] Kathleen Champion, Bethany Lusch, J. Nathan Kutz, and Steven L. Brunton. Data-driven discovery of coordinates and governing equations. *Proceedings of the National Academy of Sciences*, 116(45):22445–22451, October 2019.

[258] Urban Fasel, J. Nathan Kutz, Bingni W. Brunton, and Steven L. Brunton. Ensemble-SINDy: Robust sparse model discovery in the low-data, high-noise limit, with active learning and control. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 478(2260):20210904, April 2022.

[259] Vrushabh S. Donge, Bosen Lian, Frank L. Lewis, and Ali Davoudi. Accelerated reinforcement learning via dynamic mode decomposition. *IEEE Transactions on Control of Network Systems*, 10(4):2022–2034, December 2023.

[260] Vrushabh S. Donge, Bosen Lian, Frank L. Lewis, and Ali Davoudi. Data-efficient reinforcement learning for complex nonlinear systems. *IEEE Transactions on Cybernetics*, 54(3):1391–1402, March 2024.

Appendix A

Copies of the Papers in Original Format

A.1 An Online Reinforcement Learning Wing-Tracking Mechanism for Flexible Wing Aircraft

This is the accepted version of the paper. Published paper DOI: 10.1109/ROSE.2019.8790425.©2019 IEEE. Reprinted, with permission, from An Online Reinforcement Learning Wing-Tracking Mechanism for Flexible Wing Aircraft, IEEE International Symposium on Robotics and Sensors Environments, 06/2019.

An Online Reinforcement Learning Wing-Tracking Mechanism for Flexible Wing Aircraft

Mohammed Abouheaf, Nathaniel Mailhot, and Wail Gueaieb

Abstract— The weight shift control problem of the flexible wing aircraft is gaining increasing interest due to their unique low cost operation features and setups. The consecutive variations in the aerodynamics of the wing and additionally the kinematic constraints that evolve due to the wing-fuselage interactions make the modeling task ultimately challenging. An online adaptive control mechanism based on two linear actuation systems is proposed to fulfill different pitch-roll maneuvers. This mechanism employs model-free tracking control strategies and utilizes a real-time reinforcement learning process based on a value iteration process. The adaptation of the tracking control gains is accomplished online using the means of approximating neural network structures.

I. Introduction

Flexible wing aircraft are modeled as two-mass systems, where the fuselage and wing of the aircraft may separately rotate in pitch and roll axis relative to the other body. Hang gliders and flexible wing vehicles share common modelling considerations, where both masses are coupled via different kinematic and dynamic constraints [1]–[5], which include the kinematic constraint of the two body connection point [6], [7]. Herein, a full online automatic trajectory-tracking mechanism is developed for the flexible wing aircraft. This mechanism is based on an innovative online adaptive learning approach that decides the optimal tracking control gains in real-time with no prior knowledge about the dynamics of the vehicle.

Aerodynamic modelling of flexible wing aircraft has been explored within theoretical and experimental frameworks introduced in [1]–[5], [8], [9]. The modelling approaches employed for the aerodynamics of flexible wing's have a range varying complexity, such as simplified lateral and longitudinal decoupled dynamic equations [10], approximate rigid wing modelling with a consideration for aerodynamic derivatives [11], [12], a comprehensive decoupled aerodynamic model [13], and a nine degree of freedom aerodynamic model that employs a set of nonlinear state equations [6], [7]. During flight, the primary method for maneuvering of the flexible wing aircraft, coined weight shift control, is performed through a process of manipulating the location of the center of gravity (CoG) of the fuselage relative to the wing [2], [3], [5], [14]–[16]. The shifted fuselage CoG becomes misaligned with the center of lift of the wing, modifying the wing's shape and corresponding lift profile, leading to imbalanced aerodynamic forces acting on the wing, and eventually maneuvering of the aircraft. The longitudinal stability of flexible wing aircraft is magnified when weight shift control is employed to maintain stability [11], [12], [14]. It has been found that the lateral stability margins of flexible wing aircraft are significantly larger than compared to conventional fixed wing aircraft [5].

The dynamic programming problems suffered from difficulties related to the computational complexities of the state and action spaces which are characterized by the curse of dimensionality [17], [18]. These difficulties are solved using the Approximate Dynamic Programming (ADP) approaches [17], [19], which enable various forms of temporal difference solutions. The optimal control theory is used to solve the dynamic programming problems where the performance is optimized using utility functions [20]–[22]. The solution of the Hamilton-Jacobi-Bellman equation or the Bellman optimality equation results in solutions for the underlying optimal control problems either for single or multi-agent systems. The solution of the resulting temporal difference equations depends only on the state or state-action combinations and it can allow gradient-based solving structures. The game theory frame work is used to prepare the mathematical frame work for the multi-agent reinforcement learning problems [23], [24]. These advise coupled forms of Bellman equations or the underlying HJB equations.

Reinforcement Learning (RL) formulates a dynamic learning environment that aims to select the optimal policy and learn the optimal solution using dynamic transition mechanisms [17], [25]–[27]. The reinforcement learning solutions employ two main steps; The first is concerned with the evaluation of the temporal difference equations and the second is related to calculating the optimal strategies. These processes can be implemented in a value or policy iteration fashion [28]–[30]. Reinforcement learning approaches are used to implement the dual heuristic dynamic programming solutions in [28], where the costate equations resulting from the Hamiltonian expressions are solved using partial knowledge about the system's dynamics. Online model-free reinforcement approaches are developed for multi-agent systems in [29], [31], where the convergence is guaranteed if the graph is strongly connected and the weighting matrices are properly chosen. The neural network structures in the

This work was supported by Ontario Centers of Excellence, University of Ottawa, and Romaeris Corporation.

Mohammed Abouheaf and Wail Gueaieb are with the School of Electrical Engineering & Computer Science, Nathaniel Mailhot is with the Department of Mechanical Engineering, University of Ottawa, Ottawa, Ontario, Canada mabouhea,nmailhot,wgueaieb@uottawa.ca

scope of adaptive critics framework are useful tools to implement the reinforcement learning solutions for single and multi-agent systems [25], [26], [32], [33]. The solution approximation for each system employs two separate neural network structures that learn the solution of the underlying temporal difference equation or Bellman equation. One of the two networks approximates the temporal difference solutions and the other approximates the optimal control strategy.

The work is organized as follows. Section 2 presents briefly the basic operation and model of the flexible wing aerial system. The combined control structure is detailed out in Section 3. The optimal control problem and the associated model-free adaptive learning process are introduced in Section 4. The neural network implementation for the online reinforcement learning approach is provided in Section 5. Section 6 shows the digital simulation outcomes.

II. Weight Shift Model

The two-mass system (wing and fuselage) attaches the two bodies by the hang block, a joint that connects the mast of the fuselage and the keel tube of the wing at a shared point called the hang point. The hang block is a mechanical joint which allows for two degrees of motion relative to each body; the wing system may roll freely about the longitudinal axis of the keel bar, and the keel bar may pitch about the axis of the mast. The pilot, who is positioned within the fuselage, modifies the wings orientation with respect to the fuselage by manipulating the control bar that is rigidly attached to the wing. To modify the vehicles orientation, the pilot applies force to the control bar to move the wing with respect to the fuselage. By the principle of weight shift, this in turn modifies the lift profile of the wing, resulting in a new orientation of the wing with respect to the fuselage.

For the system considered in this work, the following weight shift kinematic model is presented with the schematic shown in Figure 1.

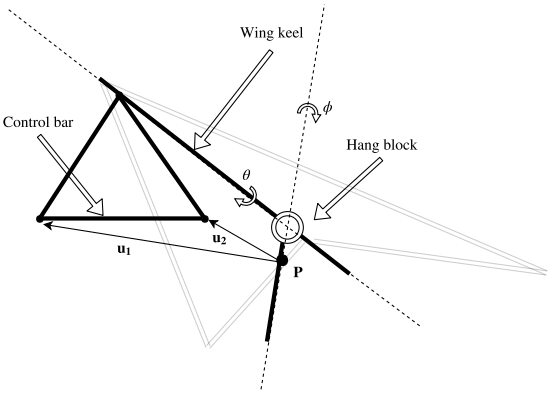


Fig. 1. Schematic for weight shift kinematics.

The orientation of the wing with respect to the fuselage is represented by the pair of rotation angles, θ and ϕ corresponding to roll and pitch of the wing. The set point of the control bar corresponds to the resting orientation of the wing in stable steady altitude, steady speed flight. The actuated motion of the control bar from the set point is represented by the pair of position vectors $\mathbf{u_1}$ and $\mathbf{u_2}$ which describe the distance between the hand grips at the two ends of the control bar, to the pilot represented by a point $\mathbf{P}$ along the mast.

The geometric relationship between the orientation of the wing with respect to the fuselage, and the change in position of the control bar is captured by expression

$$(\theta, \phi) = F(u_1, u_2),$$

which maps the wing's orientation ϕ, θ to the pair of distance vectors, $\mathbf{u_1}$ and $\mathbf{u_2}$.

To represent any mechanical force disturbances that may act to move the fuselage or wing during flight, an additional noise term is added to the roll component of wing motion, θ^{dist}, and is modelled as a normally distributed random variable with zero mean and variance of 4°.

III. The Controller Design

This section explains the mechanism of the combined navigation control scheme. As explained earlier, the navigation of the flexible wing system follows a weight shift mechanism where the center of gravity of the fuselage is changing with respect to that of the pilot to achieve the desired maneuvers. Later on, the adaptive learning solution approach is detailed out along with the neural network implementation.

The objective of the control problem is to take into account the Pitch and Roll orientations of the wing to track certain desired trajectory commands using two actuation systems with actuation lengths u_1 and u_2 (i.e., the control mechanism will decide two-optimal actuation lengths).

The developed mechanism employs two linear actuators in order to induces the required trajectory tracking responses. The combined control mechanism is shown in Figure 2, and it depends on a concurrent functioning three-tracking control loops as follows:

- First loop: (i.e., pitch controller) decides the symmetric actuation lengths u_p to achieve the desired pitching adjustment.
- Second loop: (i.e., roll controller) selects the asymmetric actuation lengths to allows the desired rolling maneuver. The control signal u_r expresses the desired asymmetry in the actuation lengths to achieve the desired roll adjustments.
- Third loop: (i.e., roll disturbance controller) considers the roll disturbances created by the fuselage-wing discrepancies while achieving the required maneuvers. The regulation signal u_d denotes the required asymmetry in the actuators lengths to suppress the

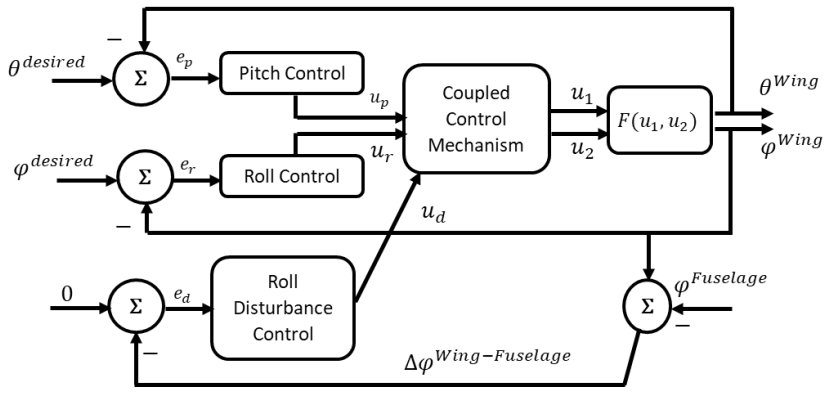


Fig. 2. Coupled Pitch-Roll Control Mechanism with Wing-Fuselage Roll Disturbances.

discrepancies between the roll angles of the wing and fuselage and reject other noise that can persist in turbulent flight conditions.

The control mechanism is designed to eliminate the tracking errors (i.e., e_p is the error between the desired pitch orientation $\theta^{desired}$ and the measured pitch orientation θ^{Wing}, e_r is the error between the desired wing roll orientation $\phi^{desired}$ and the actual wing roll orientation ϕ^{Wing}, and e_d error due to wing-fuselage disturbances). These tracking control loops employ an adaptive learning approach which utilizes the current and previous errors like PID controllers. Each roll and pitch tracking mechanism uses an adaptive learning algorithm to decide the optimal control gains.

The control laws for the tracking error control loops are given as follows

$$\begin{aligned} u_p(k) &= u_p(k-1) + \Omega_{P1}\, e_p(k) + \Omega_{P2}\, e_p(k-1) \\ &+ \Omega_{P2}\, e_p(k-2), \\ u_r(k) &= u_r(k-1) + \Omega_{R1}\, e_r(k) + \Omega_{R2}\, e_r(k-1) \\ &+ \Omega_{R2}\, e_r(k-2), \\ u_d(k) &= u_d(k-1) + \Omega_{R1}\, e_d(k) + \Omega_{R2}\, e_d(k-1) \\ &+ \Omega_{R2}\, e_d(k-2), \end{aligned} \tag{1}$$

where $\Omega_{Pi}, \forall i$ and $\Omega_{Ri}, \forall i$ are the longitudinal and lateral tracking control gains that are determined by the adaptive learning process that will be explained later on.

Remark 1: It is noted that the third control loop (i.e., disturbances regulation loop) is using the same control gains obtained for the roll tracking mechanism. This is because, the sensible disturbances happen at the roll direction and it can use the same control structure introduced by roll mechanism (i.e., the second tracking control loop).

The coupled control mechanism block shown in Figure 2 makes a logic combination of the separate selected variations in the actuation lengths u_p, u_r, and u_d. This unit builds the logic of how the actuation lengths u_1 and u_2 are inter-chosen to achieve the desired maneuvers. The resulting combined control laws are expressed as follows:

$$u_1 = u_p - (u_r + u_d)/2,$$
$$u_2 = u_p + (u_r + u_d)/2.$$

Each control law decides the symmetric (pitching) and asymmetric (rolling) fractions of the actuation movements. It is noted that the asymmetric actuation movements (u_r and u_d) cause partial pitch deviations which is corrected during the flight by the pitch control loop, which is done in a dynamic operation. Furthermore, this unit considers the minimum and maximum constraints on the allowed actuation lengths. This means that if one actuator is hitting the bounds, that over-limit actuation is compensated for by the other actuator due to the symmetric inter-distribution capability of the actuators. It is worth to mention that, u_1 and u_2 are corrected with a point in the allowed actuation length range.

The next section lays out the mathematical foundation to build the adaptive learning control mechanism.

IV. The Optimal Control Problem

This section explains the model-free Approximate Dynamic Programming Approach and develops an adaptive learning approach based on an online value iteration process. First, The temporal difference equation (Bellman equation) is stated and then the optimality principles [21], [22] are applied in order to derive the optimality conditions.

The proposed mechanism works to minimize the tracking errors (i.e., e_p, e_r and e_d) to achieve the desired mixed longitudinal-lateral maneuvers. Thus, a simple diagram for a simple tracking unit is shown in Figure 3. The signals $y^{desired}$ and $y^{measured}$ designate the desired and measured signals respectively, $e(k)$ is the tracking error signal, and $u(k)$ is the selected control signal for the dynamical system.

Herein a tracking mechanism that is similar to the basic operation of a PID controller will be introduced.

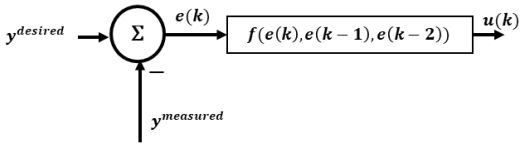


Fig. 3. The tracking control unit.

This is done using an online model-free adaptive learning mechanism. This mechanism does not need the aerodynamic model of the underlying flexible wing aircraft. Later on, this Building unit is integrated into the structured and combined control mechanism.

In this case, the main states of the adaptive learning approach are the tracking error variables (i.e., $e(k), e(k-1), e(k-2), \forall k$, k is a time-index). The objective of the optimal control problem for each building tracking unit is to select the optimal control strategy $u^o(k), \forall k$ in order to minimize a quadratic convex cost function or simply fulfill the tracking $e(k), \forall k \rightarrow 0$ objectives. Thus, the objective cost function is given by

$$C\left(\boldsymbol{E}(\ell), u(\ell)\right) = \frac{1}{2}\left(\boldsymbol{E}^T(\ell)\,\boldsymbol{Q}\,\boldsymbol{E}(\ell) + u^T(\ell)\,R\,u(\ell)\right),$$

where $E(\ell)$ is a vector of the errors so that $E(\ell) = [e^T(\ell)\ e^T(\ell-1)\ e^T(\ell-2)]^T$, $R > 0 \in R^{1\times 1}$ and $\boldsymbol{Q} \geq 0 \in R^{3\times 3}$ are symmetric weighting matrices.

This process is repeated for each tracking control loop. The utility function $C(\dots)$ is used to form the following performance index P

$$P = \sum_{\ell=0}^{\infty} C\left(\boldsymbol{E}(\ell), u(\ell)\right),$$

The control strategy $u(\ell), \forall \ell$ is selected in order to minimize the control effort and hence the performance index. The optimality conditions are found by applying Bellman optimality principles.

For simplicity $\boldsymbol{E}(\ell), u(\ell)$ are referred to as E, u for the same time index.

A quadratic value function solution form is used such that

$$S(\boldsymbol{E}, u) = \frac{1}{2}[\boldsymbol{E}^T\ \ u^T]\ \boldsymbol{\omega}\ \begin{bmatrix} \boldsymbol{E} \\ u \end{bmatrix}$$

where $\boldsymbol{\omega} = \begin{bmatrix} \boldsymbol{\omega}_{EE} & \boldsymbol{\omega}_{Eu} \\ \boldsymbol{\omega}_{uE} & \boldsymbol{\omega}_{uu} \end{bmatrix}$, $\omega_{\boldsymbol{Eu}}$, $\omega_{\boldsymbol{uE}}$, $\omega_{\boldsymbol{EE}}$, and $\omega_{\boldsymbol{uu}}$ are sub-matrix components of the matrix $\boldsymbol{\omega}$.

The performance index P and this solution form S are used to formulate the temporal difference equation or the underlying Bellman equation so that

$$S(\boldsymbol{E}(\ell), u(\ell)) = C\left(\boldsymbol{E}(\ell), u(\ell)\right) + S(\boldsymbol{E}(\ell+1), u(\ell+1)),$$

The optimaility conditions are obtained using Bellman optimality principles [20], [21] so that

$$u^o = \arg\min_{u}\left(S(\boldsymbol{E}, u)\right).$$

Then $\boldsymbol{\omega}_{uu}u + \boldsymbol{\omega}_{u\boldsymbol{E}}\boldsymbol{E} = 0$. Therefore, the model-free control strategy is given by

$$u^o = -\left[\boldsymbol{\omega}_{uu}^{-1}\,\boldsymbol{\omega}_{u\boldsymbol{E}}\right]\boldsymbol{E}. \tag{2}$$

Hence, the Bellman optimality equation follows the following form

$$\begin{aligned} S^o(\boldsymbol{E}(\ell), u^o(\ell)) &= C\left(\boldsymbol{E}(\ell), u^o(\ell)\right) \\ &+ S^o(\boldsymbol{E}(\ell+1), u^o(\ell+1)), \end{aligned} \tag{3}$$

Solving (3) using the optimal control strategy (2) guarantees a solution for the underlying optimal control problem (i.e., stabilizing the tracking errors $e(\ell), \forall \ell$).

A. Value Iteration Process

An adaptive learning process is proposed to solve all the Bellman optimality equations (3) (i.e. Bellman equations for all the tracking control loops) using the optimal control strategies (2). Value iteration is an adaptive learning process that is concerned with solving the temporal difference equation or the underlying Bellman optimality equation (3) [17], [32]–[34]. This process is composed of two main steps; First, value function evaluation. Second, control strategy improvement.

The value iteration algorithm is arranged as follows:

Algorithm 1: Value Iteration Solution

1) Initialize the solution $S^0(\dots)$ and u^0.
2) Evaluate the new solution matrix $\omega^{r+1}, \forall r$ following

$$\begin{aligned} S^{r+1}(\boldsymbol{E}(\ell), u(\ell)) &= C^r\left(\boldsymbol{E}(\ell), u(\ell)\right) \\ &+ S^r(\boldsymbol{E}(\ell+1), u(\ell+1)), \end{aligned} \tag{4}$$

where r is the value iteration solution index.
3) Evaluate improvement in the control strategy following

$$u^{r+1} = -\left[\boldsymbol{\omega}_{uu}^{-1}\boldsymbol{\omega}_{u\boldsymbol{E}}\right]^{r+1}\ \ \boldsymbol{E}. \tag{5}$$

4) Continue with the process till the differences $\Delta\boldsymbol{\omega}^{r-1,r} = \boldsymbol{\omega}^r - \boldsymbol{\omega}^{r-1}$ converge (i.e., the errors are less than a threshold) for a predefined number of iterations N.

Remark 2: This algorithm is implemented in real-time and it employs a model-free control strategy. It runs concurrently for the different scheduled control loops as will be explained later on.

The following section shows the implementation of this Reinforcement Learning algorithm using the neural network approximations.

V. Neural Network Solution Mechanism

This section explains how the adaptive learning mechanism proposed by Algorithm 1 can be implemented using the adaptive critics or the neural network structures [25], [29], [30], [32], [35]. The solving value function $S(\dots)$ and the associated control strategy $u(\dots)$ are approximated using separate neural network structures. This process enables capturing the optimality in the dynamic learning environment, where the taken strategies are rewarded or

penalized based on the accumulated cost values. The adaptation of the weights is done in real-time using simple gradient descent approach and linear quadratic cost functions are employed.

The solving value $S(\dots)$ is approximated by the following neural network structure

$$\hat{S}(\boldsymbol{E},\hat{u}) = \frac{1}{2}[\boldsymbol{E}^T \quad \hat{u}^T] \ \boldsymbol{W^T} \ \begin{bmatrix} \boldsymbol{E} \\ \hat{u} \end{bmatrix},$$

where W^T are the approximation weights for the solving value function S.

Consequently, the associated optimal control strategy $u(\dots)$ is approximated using the following neural network structure

$$\hat{u} = \boldsymbol{\Omega^T E},$$

where Ω^T are the approximation weights for the optimal control strategy.

The solving value function's approximation error ε_W is given by

$$\varepsilon_W = \left(\hat{S}(\boldsymbol{E},\hat{u}) - \tilde{S}\right),$$

where the value $\tilde{S}$ is given by

$$\tilde{S} = C\left(\boldsymbol{E}(\ell),\hat{u}(\ell)\right) + S(\boldsymbol{E}(\ell+1),\hat{u}(\ell+1)),$$

Hence, the update rule for the approximation weights W^T is given by

$$\boldsymbol{W}^{(r+1)T} = \boldsymbol{W}^{(r)T} - \eta \left(\left(\ \hat{S}(\boldsymbol{E},\hat{u}) - \tilde{S} \right) \begin{bmatrix} \boldsymbol{E} \\ \hat{u} \end{bmatrix} [\boldsymbol{E}^T \quad \hat{u}^T] \right)^{(r)} \tag{6}$$

where $0 < \eta \leq$ is a learning rate.

The gradient descent approach is used due to the simplicity of the linear quadratic cost function $C(\dots)$ which enables this direct formalization.

In a similar fashion, the approximation error for the control strategy approximation is given by $\varepsilon_u = (\hat{u} - \tilde{u})$ where

$$\tilde{\boldsymbol{u}} = -\left[\boldsymbol{\Omega}_{\hat{u}\hat{u}}^{-1}\boldsymbol{\Omega}_{\hat{u}\boldsymbol{E}}\right] \quad \boldsymbol{E}.$$

Similarly, the adaptation rule for the control strategy approximation weights follows

$$\boldsymbol{\Omega}^{(r+1)T} = \boldsymbol{\Omega}^{(r)T} - \mu \left((\hat{u} - \tilde{\boldsymbol{u}})\, \boldsymbol{E}^T \right)^{(r)}, \tag{7}$$

where μ is a learning rate.

A full online neural network solution algorithm can be arranged as follows

Algorithm 2: Neural Network Solution

1) Initialize the weights $W^0(\dots)$ and Ω^0.
2) Start with arbitrary small error values $e(-2), e(-1)$ and $e(0)$.
3) Calculate the approximation $\hat{u}$.
4) Compute $\hat{\boldsymbol{S}}$.
5) Evaluate the measurement $e(r)$ and arrange the errors' vector $\boldsymbol{E}$.
6) Calculate the approximation $\hat{u}$ and $\hat{\boldsymbol{S}}$ at iteration r.
7) find $\tilde{\boldsymbol{S}}$.
8) Evaluate the new solution value approximation weights using (6).
9) Decide the improvement in the control strategy using (7).
10) Continue with the process till the differences $\Delta \boldsymbol{W}^{r-1,r} = \boldsymbol{W}^r - \boldsymbol{W}^{r-1}, \forall r$ and $\Delta \boldsymbol{\Omega}^{r-1,r} = \boldsymbol{\Omega}^r - \boldsymbol{\Omega}^{r-1}, \forall r$ converge (i.e., the errors are less than a threshold) for a predefined number of iterations N.

The following section shows the simulation outcomes for the combined control structure in react to different desired maneuvers.

VI. Simulation Results

In this section, the simulation is performed for the combined control strategy to prove its usefulness. First, the adaptive learning algorithm is applied to the online decoupled maneuvers to advise the control gains. This is followed by a planned pitch-roll combined maneuver to show the actuation performance in order to achieve the desired maneuver. The learning rates are selected so that $\mu = \eta = 0.001$.

The simulation is performed for decoupled longitudinal maneuver to track desired pitch angles of the values -15 deg to 15 deg with 5 deg steps. The adaptations of the control strategy gains are shown in Figure (4)(a). It can be seen that the control gains are properly selected (i.e, quickly converging) in few evaluation steps. The wing orientations as shown in Figure (4)(c) in-react to the actuation lengths plotted in Figure (4)(b) ensure the proper performance of the developed controller. This is further emphasized by the reaction of the controller to a decoupled lateral maneuvers -20 deg to 20 deg with 5 deg steps. The adaptations of the controller weights are shown to converge quickly as explained by Figure (5)(a). The actuation lengths are the same since they are moving symmetrically Figure (5)(b), to achieve the desired maneuver plotted in Figure (5)(c). It is worth to note that the asymmetry in the lateral actuation motion creates fractional pitch adjustments. These adjustments are compensated for, automatically, using the pitch control Mechanism.

The combined trajectory-tracking control mechanism is tested using the predefined three control loops working all together simultaneously, with the following control gains $\Omega_p = [0.0251 \quad -0.0320 \quad -0.0139]$ and $\Omega_R = [0.0088 \quad -0.0319 \quad -0.0102]$. The desired decoupled maneuvers are formed of two turns in each direction $\theta^{desired}(r) = -10 * sin(2 * pi * r/1500)$ deg and $\phi^{desired}(r) = 10 * sin(2 * pi * r/1500)$ deg and they are happening concurrently. The wing-fuselage disturbances are modeled using stochastic noise from a normal distribution with amplitude $\pm 4deg$. Figure (6) shows the different actuation lengths provided by the control loops to achieve the desired maneuvers. The direct actuation lengths applied to the flexible wing system are shown in

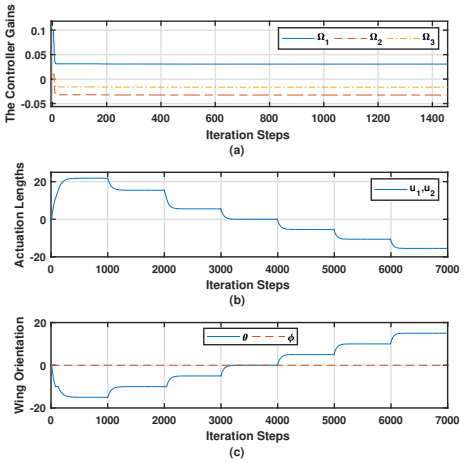


Fig. 4. Longitudinal motion. (a) The adaptation of the controller's gains. (b) The actuation lengths in response to desired maneuver. (c) The resulting wing orientation.

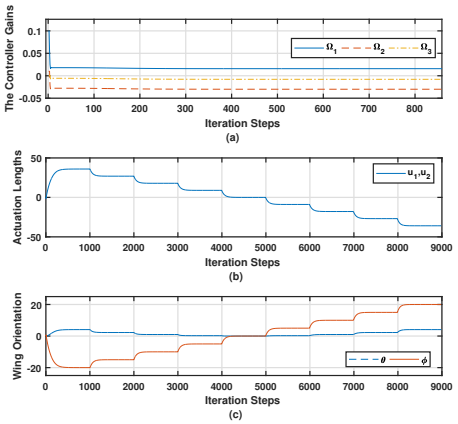


Fig. 5. Lateral motion. (a) The adaptation of the controller's gains. (b) The actuation lengths in response to desired maneuver. (c) The resulting wing orientation.

Figure (7)(a), while the actual performance compared to the desired trajectory is plotted in Figure (7)(b). These figures emphasize the ability of the combined control mechanism to satisfy the desired tracking objectives with very small lag using the online model-free adaptive learning mechanism.

VII. Conclusion

This paper introduced a novel combined trajectory-tracking control mechanism for a flexible wing system. The controller employed three interactive control loops to account for the interactions between the pilot and wing systems to achieve the desired pitch-roll tracking

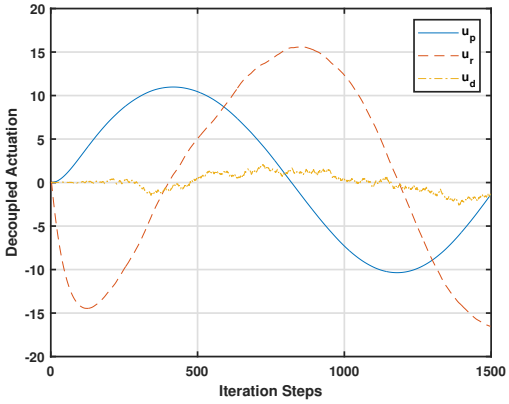


Fig. 6. The unadjusted actuation lengths by the different controllers.

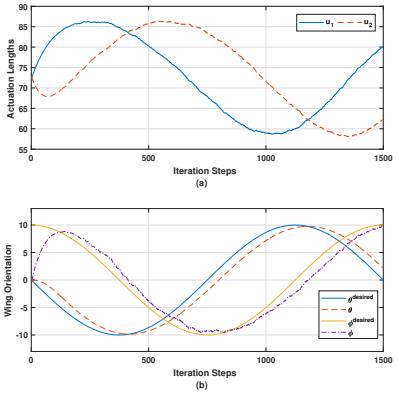


Fig. 7. Dynamic response of the coupled maneuver. (a) The actuation lengths. (b) The coupled longitudinal-lateral maneuvers.

objectives. An online adaptive learning tracking mechanism based on a value iteration process is employed to select the control gains of the different control loops. This is done with no use of any information about the aerodynamics of the flexible wing system. The solution of the adaptive algorithm is implemented and approximated using separate online neural networks. The simulation results show the effectiveness of the proposed platform in different trajectory-tracking simulation cases.

References

[1] D. Blake, "Modelling The Aerodynamics, Stability and Control of The Hang Glider," Master's thesis, Centre for Aeronautics - Cranfield University, 1991.

[2] E. A. Kilkenny, "An evaluation of a mobile aerodynamic test facility for hang glider wings," College of Aeronautics, Cranfield Institute of Technology, Tech. Rep. 8330, 1983.

[3] E. Kilkenny, "Full scale wind tunnel tests on hang glider pilots," Cranfield Institute of Technology, College of Aeronautics, Department of Aerodynamics, Tech. Rep., 1984.

[4] E. A. Kilkenny, "An experimental study of the longitudinal aerodynamic and static stability characteristics of hang gliders," phdthesis, Cranfield University, Sep. 1986.

[5] M. V. Cook, "The Theory of the Longitudinal Static Stability of the Hang-glider," The Aeronautical Journal, vol. 98, no. 978, pp. 292–304, 1994.

[6] Y. Ochi, "Modeling of the longitudinal dynamics of a hang glider," in AIAA Modeling and Simulation Technologies Conference. American Institute of Aeronautics and Astronautics, 2015, pp. 1591–1608.

[7] ——, "Modeling of flight dynamics and pilot's handling of a hang glider," in AIAA Modeling and Simulation Technologies Conference. American Institute of Aeronautics and Astronautics, 2017, pp. 1758–1776.

[8] J. Sweeting, "An Experimental Investigation of Hang Glider Stability," Master's thesis, College of Aeronautics, Cranfield University, 1981.

[9] M. Cook, 'Flight Dynamics Principles. Butterworth-Heinemann, London, 2012.

[10] I. Kroo, Aerodynamics, Aeroelasticity and Stability of Hang Gliders. Stanford University, 1983.

[11] G. De Matteis, "Response of Hang Gliders to Control," The Aeronautical Journal, vol. 94, no. 938, pp. 289–294, 1990.

[12] G. D. Matteis, "Dynamics of hang gliders," Journal of Guidance Control and Dynamics, vol. 14, no. 6, pp. 1145–1152, 1991.

[13] M. Spottiswoode, "A theoretical study of the lateral-directional dynamics, stability and control of the hang glider," mathesis, College of Aeronautics, Cranfield Institute of Technology, 2001.

[14] M. V. Cook and M. Spottiswoode, "Modelling the flight dynamics of the hang glider," The Aeronautical Journal, vol. 109, no. 1102, pp. I–XX, 2005.

[15] M. V. Cook and E. A. Kilkenny, "An experimental investigation of the aerodynamics of the hang glider," in Proceedings of an International Conference on Aerodynamics, 1986.

[16] M. V. Cook, Ed., Flight Dynamics Principles: A Linear Systems Approach to Aircraft Stability and Control, third edition ed., ser. Aerospace Engineering. Butterworth-Heinemann, 2013.

[17] R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction, second edition ed., ser. Second Edition. Massachusetts: MIT Press, 1998.

[18] P. Werbos, "Beyond Regression: New Tools for Prediction and Analysis in the Behavior Sciences," Ph.D. dissertation, Harvard University, 1974.

[19] P. J. Webros, "A menu of designs for reinforcement learning over time," in Neural Networks for Control, W. T. Miller, III, R. S. Sutton, and P. J. Werbos, Eds. Cambridge, MA, USA: MIT Press, 1990, pp. 67–95.

[20] R. Bellman, Dynamic Programming. Princeton University Press, 1957.

[21] F. Lewis, D. Vrabie, and V. Syrmos, Optimal Control, 3rd ed. New York, USA: John Wiley, 2012.

[22] A. Bryson, "Optimal Control-1950 to 1985," IEEE Control Systems, vol. 16, no. 3, pp. 26–33, 1996.

[23] T. Başar and G. J. Olsder, Dynamic Non-cooperative Game Theory , 2nd ed., ser. Classics in Applied Mathematics:. SIAM: Philadelphia, 1999.

[24] M. Abouheaf, F. Lewis, K. Vamvoudakis, S. Haesaert, and R. Babuska, "Multi-Agent Discrete-Time Graphical Games And Reinforcement Learning Solutions," Automatica, vol. 50, no. 12, pp. 3038–3053, 2014.

[25] B. Widrow, N. K. Gupta, and S. Maitra, "Punish/reward: Learning With a Critic in Adaptive Threshold Systems," IEEE Transactions on Systems, Man, and Cybernetics, vol. SMC-3, no. 5, pp. 455–465, 1973.

[26] P. Werbos, "Neural networks for control and system identification," in 28th Conference on Decision and Control, Dec. 1989, pp. 260–265.

[27] P. J. Webros, "Neurocontrol and supervised learning: An overview and evaluation," in Handbook of Intelligent Control: Neural, Fuzzy, and Adaptive Approaches, D. A. White and D. A. Sofge, Eds. Van Nostrand Reinhold, Jun. 1992, pp. 65–89.

[28] M. Abouheaf and F. Lewis, "Approximate Dynamic Programming Solutions of Multi-Agent Graphical Games Using Actor-critic Network Structures," in International Joint Conference on Neural Networks (IJCNN), Aug. 2013, pp. 1–8.

[29] ——, Dynamic Graphical Games: Online Adaptive Learning Solutions Using Approximate Dynamic Programming. World Scientific, 2014, ch. Chapter 1, pp. 1–48.

[30] M. Abouheaf, F. Lewis, M. Mahmoud, and D. Mikulski, "Discrete-time Dynamic Graphical Games: Model-free Reinforcement Learning Solution," Control Theory and Technology, vol. 13, no. 1, pp. 55–69, 2015.

[31] M. Abouheaf and M. Mahmoud, "Policy Iteration and Coupled Riccati Solutions for Dynamic Graphical Games," International Journal of Digital Signals and Smart Systems, vol. 1, no. 2, pp. 143–162, 2017.

[32] D. Bertsekas and J. Tsitsiklis, Neuro-Dynamic Programming, 1st ed. Massachusetts: Athena Scientific, 1996.

[33] L. Busoniu, R. Babuska, and B. D. Schutter, "A Comprehensive Survey of Multi-Agent Reinforcement Learning," IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), vol. 38, no. 2, pp. 156–172, 2008.

[34] P. Werbos, Approximate Dynamic Programming for Real-time Control and Neural Modeling. Handbook of Intelligent Control, D. A. White and D. A. Sorge, Eds. New York: Van Nostrand Reinhold: Van Nostrand Reinhold, New York, 1992.

[35] D. Prokhorov and D. Wunsch, "Adaptive Critic Designs," IEEE Transactions on Neural Networks, vol. 8, no. 5, pp. 997–1007, Sep. 1997.

A.2 Guidance Mechanism for Flexible Wing Aircraft Using Measurement-Interfaced Machine Learning Platform

This is the accepted version of the paper. Published paper DOI: 10.1109/TIM.2020.2985553. ©2020 IEEE. Reprinted, with permission, from Guidance Mechanism for Flexible Wing Aircraft Using Measurement-Interfaced Machine Learning Platform, IEEE Transactions on Instrumentation and Measurement, 07/2020.

Guidance Mechanism for Flexible Wing Aircraft Using Measurement-Interfaced Machine Learning Platform

Mohammed Abouheaf, Nathaniel Mailhot, Wail Gueaieb, and Davide Spinello

Abstract— The autonomous operation of flexible-wing aircraft is technically challenging and has never been presented within literature. The lack of an exact modeling framework is due to the complex nonlinear aerodynamic relationships governed by the deformations in the flexible-wing shape, which in turn complicates the controls and instrumentation setup of the navigation system. This urged for innovative approaches to interface affordable instrumentation platforms to autonomously control this type of aircraft. This work leverages ideas from instrumentation and measurements, machine learning, and optimization fields in order to develop an autonomous navigation system for a flexible-wing aircraft. A novel machine learning process based on a guiding search mechanism is developed to interface real-time measurements of wing-orientation dynamics into control decisions. This process is realized using an online value iteration algorithm that decides on two improved and interacting model-free control strategies in real-time. The first strategy is concerned with achieving the tracking objectives while the second supports the stability of the system. A neural network platform that employs adaptive critics is utilized to approximate the control strategies while approximating the assessments of their values. An experimental actuation system is utilized to test the validity of the proposed platform. The experimental results are shown to be aligned with the stability features of the proposed model-free adaptive learning approach.

I. Introduction

Flexible-wing aircraft have been capturing increasing interests due to their relatively simple mechanical structures, flexible operation, low fabrication costs, and high payload-to-mass ratio [1]–[3]. On one side, there are no experimentally-validated dynamical models for such systems since they are characterized by highly nonlinear dynamics. Therefore, designing algorithms for the autonomous flight control of this type of aircraft is a complex process. To this end, relying on conventional methods which are based on some sort of classical mathematical models, however approximate they may be, is not an option. On another side, analytical solutions for the optimal tracking control problems often necessitate solving offline coupled differential equations simultaneously, where a subset of these equations is solved backward in-time [4]. Additionally, the complexity of the tracking control laws, resulting from adaptive systems, are not realizable or hard-to-implement using digital processing units.

In this work, a coupled instrumentation-machine learning framework is proposed to solve the guidance control problem of a class of nonlinear dynamical systems with unknown dynamics, typical for flexible-wing aircraft. This process tackles difficulties associated with developing control solutions for partially- or fully model-based approaches in uncertain dynamical environments. Further, it solves the tracking control problem in real-time with no offline or backward solution for the underlying differential equations. Simply, it provides a simplified method to compute the tracking control laws which may be both complex and hard-to-realize in digital simulation environments. The proposed scheme was tested in real-time aboard of a Raspberry Pi equipped with an Inertial Measurement Unit (IMU) consisting of accelerometers, gyroscopes and magnetometers, in order to provide accurate orientation and motion sensing measurements (i.e., for the wing of the aircraft). A reinforcement learning process based on adaptive guided search mechanism is employed to steer the motion dynamics of a mock-up emulating the aircraft's wing movement relative to the fuselage. It employs an online value iteration process that produces real-time model-free tracking control strategies.

Flexible-wing aircraft are composed of two interacting structures, namely wing and fuselage, which are pinned at a point where they can rotate on pitch-roll axes relative to one another [5]–[8]. The control of this type of aircraft is challenging due to the flexibility of the wing which induces continuous aerodynamic variations and makes it difficult to model the vehicle's dynamics [1], [9]. Partial theoretical models have been presented for this type of aircraft in [8], [10], [11]. The approximate aerodynamic models varied in complexity and approach. Some researchers opted to decouple the dynamics into longitudinal and lateral motion frames to simplify the problem [12]. A fully decoupled aerodynamic model is developed in [13]. In another context, a fixed-wing approach that considers aerodynamic derivatives is employed in [14], [15]. Additionally, a system of nonlinear dynamical equations based on a nine-degree-of-freedom dynamic model is derived in [1], [9]. The navigation process of this aircraft is based on a weight-shift mechanism where the relative variations in the Center-of-Gravity (CoG) locations of the wing-fuselage systems, under kinematic and dynamic constraints at the interconnection points, steer the aircraft toward the

This work was partially supported by Ontario Centers of Excellence (OCE). Grant number 27404.

Mohammed Abouheaf and Wail Gueaieb are with the School of Electrical Engineering & Computer Science, University of Ottawa, Ottawa, Canada. Nathaniel Mailhot and Davide Spinello are with the Department of Mechanical Engineering, University of Ottawa, Ottawa, Canada. E-mail: {mabouhea,nmailhot,wgueaieb,dspinell}@uottawa.ca

desired orientations [16]–[18]. It is shown that when a weight-shift mechanism is applied, the longitudinal stability is magnified [15], [16]. Also, the stability margins corresponding to the lateral motion are shown to be larger than those of fixed-wing aircraft [8].

The tracking, navigation, and motion-guided systems employ instrumentation configurations along with theoretical advances for the underlying physical mechanisms. A method based on radio frequency identification technology is proposed for navigating mobile robots in [19]. An optoacoustic positioning scheme that utilizes inertial and distance measurements is developed for automated monitoring of complex manual assembly operations in [20]. It employs a particle filter that fuses inertial navigation measurements with unilateral distances measurements to space-fixed receiving devices. An artificial neural network approach is proposed to fuse differential and uncompensated measurements from global position and inertial-navigation devices respectively in [21]. This approach avoided using Kalman filtering in order to integrate the inertial navigation and global position systems. Adaptive robust tracking control approach that uses a continuous model based estimator is employed to control a flexible air-breathing hypersonic vehicle in [22]. It employs a type-2 Fuzzy structure to approximate the unknown model dynamics and the stability features are validated using a Lyapunov theorem. A comprehensive test-bed that utilizes multi-camera for operations which involve control and 3D tracking tasks of unmanned aerial vehicles is developed in [23]. The embedded navigation approach relies on a Proportional-Derivative (PD) control structure that receives navigation information form multi-camera system. The test-bed allowed real-time computations at 100 Hz using cameras with field programmable gate array (FPGA) modules where the embedded software is able to perform motion control and image processing. A vision-based tracking approach that uses particle swarm optimization and fuzzy logic scheme is developed to navigate an autonomous mobile robot in [24]. The fuzzy tracking system is designed using a Lyapunov framework and it benefits global search capabilities of the particle swarm optimization technique. An approach based on interval analysis is developed to solve the localization problem of a mobile robot using ultrasonic sensors in [25]. This approach manages the issues arising from data-associate step noticed within the classical localization problems using Kalman filtering.

Dynamic Programming (DP) solution techniques are employed to solve different control problems [4], [26], [27]. However, these frameworks degrade due to the curse of dimensionality associated with the state-action domains [26], [27]. Approximate Dynamic Programming (ADP) relaxed the manner at which dynamic programming problems are solved using heuristic platforms [26], [28]. These solution forms are meant to provide computational platforms to solve the control problems using temporal difference structures [29]–[31]. The control problems are solved by optimizing the performance of the underlying dynamical systems using objective cost functions. Hence, the solutions for the underlying Bellman optimality or Hamilton-Jacobi-Bellman equations lead to solutions for the optimal control problems [4], [32], [33]. These optimal forms vary in structure, as they depend either on Bellman or Hamiltonian structures, allowing different temporal difference solution forms [34], [35]. These solutions become more complicated in case of coupled hierarchical control systems or multi-agent structures [30], [36]. This is due to the existence of coupled temporal difference formulations. The ADP problems are solved using a dynamic learning environment known as Reinforcement Learning (RL) [26], [37]–[39]. In this process, the strategies taken by the agent are either rewarded or penalized based on their value-assessments using a utility or an objective cost function. The policy-value assessment mechanism is repeated until the best strategy is found.

Reinforcement learning solutions are implemented using two-step mechanisms known as value iteration and policy iteration [26], [37]. The first step solves the temporal difference equation, while the second approximates the optimal control policy [34], [35]. In value iteration, the solving value function is evaluated and then the best strategy is extracted. While in policy iteration, a strategy is evaluated in the first step then a policy improvement step is introduced [26]. The adaptive learning solutions are developed for multi-agent systems where the communications between the nodes are accomplished using graph structures in [30], [35], [40], [41]. These solution forms were able to solve coupled temporal difference equations in real-time using only neighborhood information. Adaptive critics are used as neural network approximation tools to implement these solutions for single- and multi-agent control problems [37], [38], [42], [43]. Each adaptive critics scheme involves two approximating neural networks; the actor network approximates the optimal control strategy while the critic network approximates the solving value function. A cell-mapping approach based on a reinforcement learning technique is developed for robot motion planning in [44]. The learning mechanism does not employ a dynamical model for the robot as it builds experience-knowledge about the dynamical environment and robots dynamics. It employs a transformation based on cell-to-cell transitions in order to reduce time used to build experience about the dynamical environment.

The proposed work advances initial research investigations aimed to design intelligent flight controllers for flexible-wing aircraft using a linear actuation mechanism and machine learning process [45]. However, this development relied on a geometric model of the aircraft and the control strategies are mapped to actuation lengths. Herein, an experimental mock-up system that is composed of servo actuation winch motors acting on the mast and wing of a flexible-wing aircraft are

employed to achieve autonomous maneuvers. In [45], the desired strategies are only evaluated in terms of the tracking error signals, while the proposed mechanism finds control decisions based on tracking error signals, their dynamics, and the dynamical measures of the key orientation parameters to support the overall stability of the aircraft. Finally in contrast to [45], this work provides a flexible and innovative model-free guided search learning process based on a real-time value iteration scheme. The machine learning search method opens the door to reflect the designer considerations, regarding data driven structures, in the architecture of the learning process.

The contributions of this work are four-fold. First, an innovative experimental platform is developed for autonomous control of flexible-wing aircraft. This is known to be a complicated task which was not realized using online model-free control systems before introducing this work. Second, a new machine learning process is proposed. It uses real-time measurements and it conditions the optimization objectives in order to guide the search for the best control strategies in an online fashion with high success rate. Third, it provides ideas related to the online solutions of the optimal tracking control problems, which are solved basically in an offline fashion. This approach does not employ any model-dependent control strategies and it is easily integrated into an off-the-shelf computing unit, such as a Raspberry Pi. Finally, the research presented herein provides a flexible framework that can be generalized for controlling complex nonlinear dynamical systems of the same class as flexible-wing aircraft.

The remaining sections of the paper are arranged as follows. The operation of flexible-wing aircraft is briefly highlighted in Section II. The measurement scheme and the adopted real-time sensory devices are detailed in Section III. The different control and optimization objectives are presented in Section IV. This is followed by the development of the machine learning process and its adaptive critics implementation in Section V. The digital experimental results are analyzed in Section VI in order to evaluate the validity of the proposed control setup. Finally, conclusions are drawn in Section VII.

II. Operation of Flexible-wing Aircraft

In this section, the basic kinematic relationships concerned with pitch motion control of the flexible-wing aircraft are detailed out. The two-mass system (i.e., wing and fuselage) attaches the two bodies by a hang block, a joint that connects the mast of the fuselage and the keel tube of the wing at a common point known as the hang point. The hang block is a mechanical joint which allows for two degrees of motion relative to each body. The wing system may roll freely about the longitudinal axis of the keel tube, while the keel bar may pitch about the axis which is perpendicular to both the mast and keel tubes. During manned flight, the pilot who is positioned within the fuselage, modifies the wings orientation with respect to the fuselage by manipulating the control bar that is rigidly attached to the wing. To modify the vehicle's orientation, the pilot applies a force to the control bar to shift the wing with respect to the fuselage. By the principle of weight shift, this in turn modifies the lift profile of the wing, resulting in a new wing orientation with respect to the fuselage.

The work presented in this paper is a first phase of a long-term project aiming at devising a model-free control algorithm for a commercial flexible-wing cargo aircraft. The first phase focuses on controlling the wing's orientation with respect to the fuselage when the aircraft is either stalled on the ground or is in a taxi mode on the runway. To this end, an experimental mock-up is set up to emulate the relative actuation mechanism between the wing and fuselage. The weight-shift kinematics of this mechanism is schematically depicted in Figure 1. A pair of servomotor winches are mounted at point M on the mast, and connected to points K and K' on the keel tube. The orientation of the wing with respect to the fuselage is represented by the pair of rotation angles, θ and ϕ corresponding to pitch and roll of the wing. It is important to note that the wing does not rotate along the yaw axis, represented by ψ, as it is constrained by the hang point on that axis. The set point of the control bar corresponds to the resting orientation of the wing in stable steady altitude and steady speed flight. Pull-only forces applied to the wing by the winch servomotors are represented by the pair of position vectors f_{fore} and f_{aft}, and act in the free motion axis of pitch perpendicular to the keel tube with unknown angles proportional to $\angle KK'M$ and $\angle K'KM$, respectively. For this work, it is assumed that the wing control in the pitch axis is sufficiently decoupled from the roll axis, such that the servomotor winch actions due to their geometric configuration do not induce the roll motion of the wing.

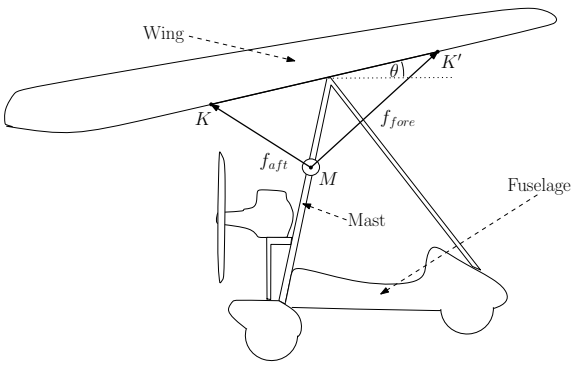


Fig. 1: Schematic of weight shift kinematics of flexible-wing aircraft with two servo winches acting on wing to affect pitch.

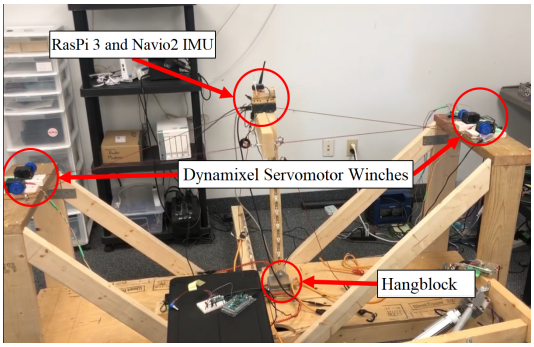


Fig. 2: Experimental mock-up used to emulate the motion of the fuselage about the wing in ground test. Note that the wing-mast system is oriented upside down; i.e., the top surface of the middle bar, representing the wing, is facing the floor, while the mast likewise sits above it. The figure also shows the Navio2+Raspi IMU and controller, servomotor winches, and hangblock.

III. Instrumentation and Measurement Platform

The experimental mock-up emulating the relative motion between the fuselage and the wing is shown in Figure 2. A block diagram of how the system's interconnected blocks are interfaced is revealed in Figure 3. Efforts were made to ensure each hardware device and supporting software is either open-source or has publicly available documentation. An Emlid Navio2 [46] board attached to a Raspberry Pi 3 [47] is selected as the controller's computational unit due its relatively convenient and easy user setup and prevalence as a hardware of choice for open-source flight controller platforms, such as ArduPilot [48] and ROS [49]. Navio2 provides redundant wing orientation estimates through the combination of sensor readings from either the main on-board InvenSense MPU-9250 9-Degree-of-Freedom (DOF) Inertial Measurement Unit (IMU) [50] or the secondary Microelectronics LSM9DS1 9 DOF IMU [51], as well as global positioning system (GPS). Each IMU has a three-axis magnetometer, three-axis gyroscope, and three-axis accelerometer. The IMU signals are filtered by a programmable digital low pass filter, which is then fed to a portable mini-computer (Raspberry Pi 3 model B). The unit has a 1.2GHz 64-bit quad-core ARMv8 CPU, 1GB of RAM and a Cortex-M3 co-processor.

Angular position measurements of the wing orientation were sampled at 20 Hz, a rate far below the maximum of 8 kHz supported by the dual IMUs. The total root mean square (RMS) noise of the MPU-9250 used for inertial measurement feedback is provided to be $0.1\,^{\circ}/\mathrm{s}$. The measurements data are filtered by a low-pass filter with a cut-off frequency of 250 Hz. The control of the platform was actuated by two Dynamixel XM-430 servomotors [52]. The servomotors receive current input commands at a rate of 20 Hz from the controller embedded within the Raspberry Pi 3. In reference to Figure 2, the system is arranged as if the aircraft wing was upside down, where the hangblock and wing are closest to the ground, and the mast sits above it. This setup was chosen for its convenience but it should have no effect on the validity of the experimental results. The mock-up allows for the mast to move relative to the wing. The servomotor winches are rigidly attached to the wing keel, and the attachment points are at opposite points along the mast. The results presented in Section VI are captured from experiments with this mock-up system where two servomotors are mounted directly to the wing. Desired reference position commands are programmed prior to the experiments. The learning algorithm decides the best control policy using IMU observations, in real time, and transmits current signals, based torque commands, to the servomotors to track the reference signal.

IV. The Control and Optimization Problem

In the following section, ideas behind the online machine learning process are first developed. Thereafter, the governing Bellman or temporal difference equations are derived. Further, this section discusses how the different control tasks (i.e., tracking and stabilization) are coordinated simultaneously in real-time.

A. Optimal Control Structure

Due to the aircraft's complex nonlinear dynamics, it is necessary to avoid building a control or computational approach that relies explicitly on existence of an aerodynamic model. As such, any control solution would better be based on robust model-free computational mechanisms.

The wing's pitch control mechanism is realized using two interacting control objectives. The first looks for a control strategy that receives real-time tracking error signals, arranged using pre-designed criteria, and provides a control signal that minimizes the tracking error. The second searches for an auxiliary control policy that supports the stability of the overall system during the maneuvers. The full autonomous control system is schematically represented in Figure 4. These control objectives are integrated together and implemented using a guided search algorithm that is developed herein based on an innovative machine learning process (as shall be discussed later).

The structure of the tracking error vector $E_\ell \in \mathbb{R}^o$ reflects the design objectives corresponding to the tracking control strategy π_E which in turn decides the tracking control signal $u_\ell^{\pi_E} \in \mathbb{R}^m$. The error vector E_ℓ relies on various dynamic forms of the tracking error signals e_ℓ (ℓ is a time-index). On another side, the stabilizing control policy π_X maps the observable key measurements in vector $X_\ell \in \mathbb{R}^n$ to a state feedback control signal $u_\ell^{\pi_X} \in \mathbb{R}^s$ in order to support the system's stability during the navigation process. The overall

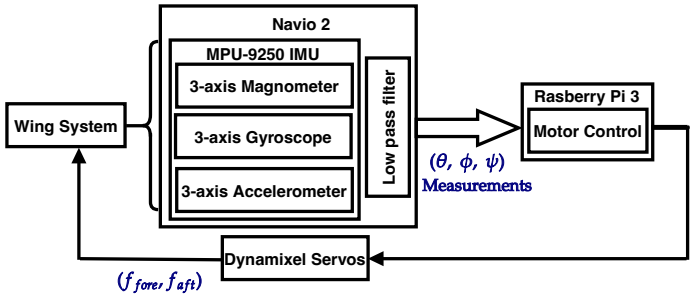


Fig. 3: Flow schematic of experimental instrumentation.

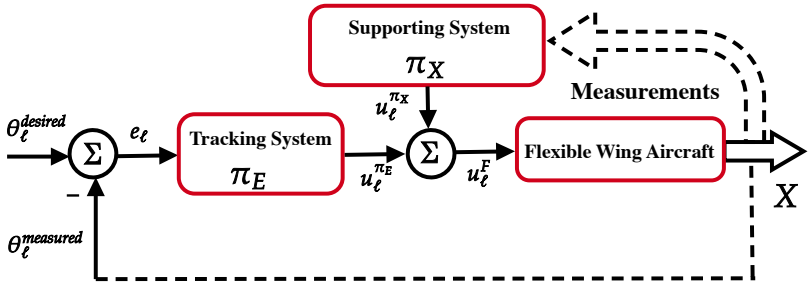


Fig. 4: Feedback control loop.

control $u_\ell^F \in \mathbb{R}^{\max\{s,m\}}$ which is applied to the actuation system, results from combining the dynamical effects from both control policies (i.e., π_E and π_X). The control signals u^{π_E}, u^{π_X}, and hence u^F, are the actuation signals responsible to move the control bar of the flexible-wing aircraft. It is worth to note that in order to generalize the proposed approach, the control signals u^{π_E} and u^{π_X} could have different vector sizes and the total control signal u^F sums the dynamical effects of the matched actuation signals in both control laws. In this work, the control signals which are fed to the actuation systems are scalars. This simplifies the form of the collective control signal u^F.

Performance indices $J^{\pi_E}(E_\ell)$ and $J^{\pi_X}(X_\ell)$ are used to asses the usefulness of the attempted policies π_E and π_X, respectively, so that

$$\begin{aligned} J^{\pi_E}(\boldsymbol{E}_\ell) &= \sum_{i=\ell}^{\infty} C^E\left(\boldsymbol{E}_i, \boldsymbol{u}_i^{\pi_E}\right) \\ J^{\pi_X}(\boldsymbol{X}_\ell) &= \sum_{i=\ell}^{\infty} C^X\left(\boldsymbol{X}_i, \boldsymbol{u}_i^{\pi_X}\right), \end{aligned} \tag{1}$$

where C^E and C^X are cost functions associated with the performance indices $J^{\pi_E}(\boldsymbol{E}_\ell)$ and $J^{\pi_X}(\boldsymbol{X}_\ell)$, respectively. They are given by

$$C^E(\boldsymbol{E}_\ell, \boldsymbol{u}_\ell^{\pi_E}) = \frac{1}{2}\left(\boldsymbol{E}_\ell^T\, \boldsymbol{Q}^E\, \boldsymbol{E}_\ell + \boldsymbol{u}_\ell^{\pi_E\,T}\, \boldsymbol{R}^E\, \boldsymbol{u}_\ell^{\pi_E}\right)$$
$$C^X(\boldsymbol{X}_\ell, \boldsymbol{u}_\ell^{\pi_X}) = \frac{1}{2}\left(\boldsymbol{X}_\ell^T\, \boldsymbol{Q}^X\, \boldsymbol{X}_\ell + \boldsymbol{u}_\ell^{\pi_X\,T}\, \boldsymbol{R}^X\, \boldsymbol{u}_\ell^{\pi_X}\right),$$

where $\boldsymbol{Q}^E \in \mathbb{R}^{o\times o}$, $\boldsymbol{R}^E \in \mathbb{R}^{m\times m}$, $\boldsymbol{Q}^X \in \mathbb{R}^{n\times n}$, and $\boldsymbol{R}^X \in \mathbb{R}^{s\times s} > 0$ are symmetric positive definite weighting matrices.

The objective functions C^E and C^X have convex forms and quadratic dependencies on the different policies and real-time measurements. These forms motivate and enable Lyapunov stability proofs for the underlying temporal difference solutions [31], [53].

Remark 1: The design of the optimization problem could variate from the one that is considered herein. This is decided by the designer which judges the different segments of the optimization problem and how they are hierarchically organized. In this development, it depends on the way key measurements are allowed and collected for the control design and on the dynamics which the controller needs to consider or regulate during the navigation process.

B. Mathematical Solution Framework

The control solution developed herein maps the optimization objectives mentioned above into temporal difference forms using a discrete-time optimal control framework [4]. Quadratic forms of solving value functions $V^{\pi_E}(\boldsymbol{E}_\ell)$ and $V^{\pi_X}(\boldsymbol{X}_\ell)$ are advised to evaluate the quality of the computed control strategies. They are motivated based on the structures of the indices $J^{\pi_E}(\boldsymbol{E}_\ell)$ and $J^{\pi_X}(\boldsymbol{X}_\ell)$ and the associated cost functions C^E and

C^X, respectively, such that

$$V^{\pi_E}(\boldsymbol{E}_\ell) = \frac{1}{2}\boldsymbol{E}_\ell^T\, \boldsymbol{S}^E\, \boldsymbol{E}_\ell \quad \equiv \quad J^{\pi_E}(\boldsymbol{E}_\ell)$$
$$V^{\pi_X}(\boldsymbol{X}_\ell) = \frac{1}{2}\boldsymbol{X}_\ell^T\, \boldsymbol{S}^X\, \boldsymbol{X}_\ell \quad \equiv \quad J^{\pi_X}(\boldsymbol{X}_\ell),$$

where $S^E \in \mathbb{R}^{o\times o}$ and $S^X \in \mathbb{R}^{n\times n}$ are solution matrices. They play a major role in the guided search policies found using the online adaptive learning process.

The matrices S^E and S^X are computed using a temporal difference solution framework in real-time, and hence are relevant to the choice of the best policies π_E and π_X, respectively, using the interactive learning process.

The value functions V^{π_E} and V^{π_X} are utilized to form Bellman equations (temporal difference solution structures) for the underlying control mechanism, such that

$$V^{\pi_E}(\boldsymbol{E}_\ell) = C^E(\boldsymbol{E}_\ell, \boldsymbol{u}_\ell^{\pi_E}) + V^{\pi_E}(\boldsymbol{E}_{\ell+1})$$
$$V^{\pi_X}(\boldsymbol{X}_\ell) = C^X(\boldsymbol{X}_\ell, \boldsymbol{u}_\ell^{\pi_X}) + V^{\pi_X}(\boldsymbol{X}_{\ell+1}). \qquad (2)$$

These Bellman equations indicate that two interacting optimization processes are solved simultaneously, where it is required to drive the tracking error $\boldsymbol{E}$ to zero and, at the same time, optimize the dynamics $\boldsymbol{X}$ along the navigation trajectories. Further, Bellman equations (2) enable the integration between two environments. The first is related to solving the navigation control problem, while the second enables an approximate dynamic programming solution (i.e., machine learning solution) for the problem. The online adaptive learning control process starts with an initial control strategy for each control decision and then the learning process, employing the above Bellman equations, directs the control strategies (i.e., learn better control decisions) using a value iteration process which is guaranteed to converge.

Herein, the error vector $\boldsymbol{E}_\ell$ is structured as follows:

$$\boldsymbol{E}_\ell = \begin{bmatrix} e_\ell & e_{\ell-1} & e_{v\ell} & e_{v\ell-1} & e_{s\ell} & s_{s\ell-1} \end{bmatrix}^T$$

where $e_{v\ell}$ and $e_{v\ell-1}$ are error derivatives, with respect to time, evaluated at time ℓ and $\ell-1$, respectively. $e_{s\ell}$ and $s_{s\ell-1}$ are the moving averages calculated as $e_{s\ell} = \frac{1}{N}\sum_{i=\ell-N}^{\ell} e_i$ and $e_{s\ell-1} = \frac{1}{N}\sum_{i=\ell-N-1}^{\ell-1} e_i$, respectively.

The way the error vector $\boldsymbol{E}_\ell$ is calculated, combines many sub-objectives which are optimized together. It minimizes the local tracking error e_ℓ while considering its previous instance $e_{\ell-1}$. Further, it smoothens the control decision by looking backward in-time to include error derivatives evaluated at time instances ℓ and $\ell-1$, as well as evaluating average of the errors across longer time-intervals N. It is worth to mention that the vector formulation $\boldsymbol{E}_\ell$ enables more advanced forms of equivalent discrete Proportional-Integral-Derivative (PID) structures (i.e., dependence on $e_\ell, e_{\ell-1}$, and $e_{\ell-2}$) or even considers higher-order derivatives and integral equivalents (i.e., dependence on $e_\ell, e_{\ell-1}, e_{\ell-2}, \ldots$, and $e_{\ell-N}$).

The solution of the optimal tracking problem relies on solving a number of differential equations, a subgroup of which is solved offline and at the same time they do not allow dynamical forms of the error signals. Herein, the formulation of the adaptive learning mechanism allows for an online solution as well as using a variety of tracking error dynamical forms.

The experimental setup, described in Section III, provides measurements related to the aerodynamic orientation of the wing. This made it convenient to choose the states $\boldsymbol{X}_\ell$ as

$$\boldsymbol{X}_\ell = \begin{bmatrix} \theta_\ell & \theta_{v\ell} & \theta_{a\ell} \end{bmatrix}^T,$$

where $\theta_\ell, \theta_{v\ell}$, and $\theta_{a\ell}$ are the pitch attitude, pitch velocity, and pitch acceleration, respectively.

The desired control policies $\boldsymbol{\pi}_E$ and $\boldsymbol{\pi}_X$ are linear feedback polices which are used to decide on the different control signals, such that

$$\boldsymbol{u}_\ell^{\pi_E} = \boldsymbol{\pi}_E\, \boldsymbol{E}_\ell, \quad \boldsymbol{u}_\ell^{\pi_X} = \boldsymbol{\pi}_X\, \boldsymbol{X}_\ell, \qquad (3)$$

Next, we will explain how to compute the real-time negative feedback control strategies $\boldsymbol{\pi}_E$ and $\boldsymbol{\pi}_X$.

V. Machine Learning Platform

The online computational machine learning framework is driven by the design of the control problem. First, we will present a computational platform based on an innovative value iteration implementation for an online reinforcement learning solution. Then, adaptive critics are employed to provide neural network implementation for the approximate dynamic programming solution.

A. Guided Search Process

The policies $\boldsymbol{\pi}_E$ and $\boldsymbol{\pi}_X$ are guided toward the intended optimization objectives stated by the designer as follows:

$$\boldsymbol{u}^{\pi_E} = -\boldsymbol{P}^E \frac{\partial V^{\pi_E}(\boldsymbol{E}_\ell)}{\partial \boldsymbol{E}_\ell}, \quad \boldsymbol{u}^{\pi_X} = -\boldsymbol{P}^X \frac{\partial V^{\pi_X}(\boldsymbol{X}_\ell)}{\partial \boldsymbol{X}_\ell}, \quad (4)$$

where $\boldsymbol{P}^E \in \mathbb{R}^{m\times o}$ and $\boldsymbol{P}^X \in \mathbb{R}^{s\times n}$ are guiding search vectors which carry the intentions and dynamic preferences of the control mechanism designer.
The vectors $\boldsymbol{P}^E$ and $\boldsymbol{P}^X$ contain selective dynamic forms (i.e., introduced by the designer to reflect their objectives regarding the entries in vectors $\boldsymbol{E}$ and $\boldsymbol{X}$) as will be highlighted in the experimental analysis. This in turn promotes flexibility for a new class of online learning processes with guided search features.

The control polices $\boldsymbol{\pi}_E = -\boldsymbol{P}^E\, \boldsymbol{S}^E$ and $\boldsymbol{\pi}_X = -\boldsymbol{P}^X\, \boldsymbol{S}^X$ (i.e., these associated with $\frac{\partial V^{\pi_E}(\boldsymbol{E}_\ell)}{\partial \boldsymbol{E}_\ell}$ and $\frac{\partial V^{\pi_X}(\boldsymbol{X}_\ell)}{\partial \boldsymbol{X}_\ell}$) are applied in real-time using a value iteration process as will be explained next.

B. Value Iteration Algorithm

An online reinforcement learning solution is developed using the aforementioned Bellman equations (2). The solution is realized using a two-step value iteration process. The first updates the solving value functions (i.e., $\boldsymbol{S}^E$ and $\boldsymbol{S}^X$) using (2) while the second extracts the new or improved policies (i.e., $\boldsymbol{\pi}_E$ and $\boldsymbol{\pi}_X$). The online value iteration process is detailed out in Algorithm 1. Bellman equations provide the temporal difference platform necessary to guide the online learning process beyond the initially selected polices $\boldsymbol{\pi}_E^0$ and $\boldsymbol{\pi}_X^0$. Note that, unlike policy iteration paradigms, these initial policies do not need to be admissible. Hence, the intentionally guided policies are improved along the trajectory of the aircraft.

Remark 2: Value iteration processes are proven to converge in general for single and multi-agent systems based on Lyapunov stability approaches if the underlying systems are stabilizable [30], [53], [54]. This is mainly due to the properties of convex objective cost functions (i.e., C^E and C^X in this work) and consequently Bellman solution forms (i.e., (2)). Hence, the value functions evolve in a bounded manner, such that

$$\begin{aligned} 0 \leq V^{\pi_E(0)}(\boldsymbol{E}_\ell) \leq V^{\pi_E(1)}(\boldsymbol{E}_\ell) \leq \cdots \leq \\ V^{\pi_E(t)}(\boldsymbol{E}_\ell) \leq \cdots \leq V^{\pi_E(*)}(\boldsymbol{E}_\ell), \\ 0 \leq V^{\pi_X(0)}(\boldsymbol{X}_\ell) \leq V^{\pi_X(1)}(\boldsymbol{X}_\ell) \leq \cdots \leq \\ V^{\pi_X(t)}(\boldsymbol{X}_\ell) \leq \cdots \leq V^{\pi_X(*)}(\boldsymbol{X}_\ell), \end{aligned} \tag{5}$$

where the value functions $V^{\pi_E(*)}$ and $V^{\pi_X(*)}$ are the optimal response solutions for Bellman equations (2). In other words, the tracking error and motion dynamics are stable if the aircraft system is controllable.

The online value iteration learning process guides the solving value functions toward the best values $V^{\pi_E(*)}(\boldsymbol{E}_\ell)$ and $V^{\pi_X(*)}(\boldsymbol{X}_\ell)$, and hence best guided policies $\boldsymbol{\pi}_E^*$ and $\boldsymbol{\pi}_X^*$.

C. Adaptive Critics Implementation

Adaptive critics are employed as neural network approximation tools for the introduced online reinforcement learning solution [37]. The adaptive critics scheme is implemented using means of neural network structures, namely actor-critic networks, for each optimization control process. They provide guided search solutions for the underlying Bellman equations. The critic and actor structures approximate the solving value functions and the associated policies in an interactive manner implicitly during the iterative update of underlying Bellman equations. The actor network reflects the improvements in the guided control strategy while its quality is approximated by the critic network. The weights of the actor-critic structures are adapted using a gradient decedent approach motivated by the linear forms of the control policies.

The value functions V^{π_E} and V^{π_X} (i.e., the critic structures) are approximated so that

$$\hat{V}^{\pi_E}(\boldsymbol{E}_\ell) = \frac{1}{2}\boldsymbol{E}_\ell^T\,\boldsymbol{\Omega}_c^E\,\boldsymbol{E}_\ell, \hat{V}^{\pi_X}(\boldsymbol{X}_\ell) = \frac{1}{2}\boldsymbol{X}_\ell^T\,\boldsymbol{\Omega}_c^X\,\boldsymbol{X}_\ell,$$

where $\boldsymbol{\Omega}_c^E \in \mathbb{R}^{o\times o}$ and $\boldsymbol{\Omega}_c^X \in \mathbb{R}^{n\times n}$ are the critic approximation weights of the value functions $\hat{V}^{\pi_E}(\boldsymbol{E}_\ell)$ and $\hat{V}^{\pi_X}(\boldsymbol{X}_\ell)$, respectively. The critic network forms are motivated by the structures of the value functions V^{π_E} and V^{π_X} respectively.

In a similar fashion, the guided search polices (i.e., the actor structures) are approximated so that

$$\hat{\boldsymbol{u}}_\ell^{\pi_E} = \boldsymbol{\Omega}_a^E\boldsymbol{E}_{\boldsymbol{\ell}}, \quad \hat{\boldsymbol{u}}_\ell^{\pi_X} = \boldsymbol{\Omega}_a^X\boldsymbol{X}_{\boldsymbol{\ell}},$$

where $\boldsymbol{\Omega}_a^E \in \mathbb{R}^{m\times o}$ and $\boldsymbol{\Omega}_a^X \in \mathbb{R}^{s\times n}$ are the actor approximation weights of the policies $\boldsymbol{\pi}_E$ and $\boldsymbol{\pi}_X$, respectively.

The different approximation weights are updated using a gradient descent approach that applies a minimization criteria on the squared approximation errors. The approximation errors of the critic networks have squared forms, so that

$$\begin{aligned} \varepsilon_c^{E_\ell} &= \frac{1}{2}\left(\hat{V}^{\pi_E}(\boldsymbol{E}_\ell) - \hat{V}^{T_E}(\boldsymbol{E}_\ell)\right)^2, \\ \varepsilon_c^{X_\ell} &= \frac{1}{2}\left(\hat{V}^{\pi_X}(\boldsymbol{X}_\ell) - \hat{V}^{T_X}(\boldsymbol{X}_\ell)\right)^2, \end{aligned} \tag{6}$$

where the target values $\hat{V}^{T_E}(\boldsymbol{E}_\ell)$ and $\hat{V}^{T_X}(\boldsymbol{X}_\ell)$ are calculated using

$$\begin{aligned} \hat{V}^{T_E}(\boldsymbol{E}_\ell) &= C^E(\boldsymbol{E}_\ell, \hat{\boldsymbol{u}}_\ell^{\pi_E}) + \hat{V}^{\pi_E}(\boldsymbol{E}_{\ell+1}) \\ \hat{V}^{T_X}(\boldsymbol{X}_\ell) &= C^X(\boldsymbol{X}_\ell, \hat{\boldsymbol{u}}_\ell^{\pi_X}) + \hat{V}^{\pi_X}(\boldsymbol{X}_{\ell+1}). \end{aligned}$$

The critic weights are adapted according to a gradient descent rule such that

$$\begin{aligned} \boldsymbol{\Omega}_c^{E(t+1)} &= \boldsymbol{\Omega}_c^{E(t)} - \alpha_c^E\left[\frac{\partial\varepsilon_c^{E_\ell}}{\partial\boldsymbol{\Omega}_c^E}\right]^{(t)} \\ \boldsymbol{\Omega}_c^{X(t+1)} &= \boldsymbol{\Omega}_c^{X(t)} - \alpha_c^X\left[\frac{\partial\varepsilon_c^{X_\ell}}{\partial\boldsymbol{\Omega}_c^X}\right]^{(t)}, \end{aligned} \tag{7}$$

where $0 < \alpha_c^E, \alpha_c^X < 1$ are critic networks learning rates, $\frac{\partial\varepsilon_c^{E_\ell}}{\partial\boldsymbol{\Omega}_c^E} = \left(\hat{V}^{\pi_E}(\boldsymbol{E}_\ell) - \hat{V}^{T_E}(\boldsymbol{E}_\ell)\right)\boldsymbol{E}_\ell\,\boldsymbol{E}_\ell^T$, $\frac{\partial\varepsilon_c^{X_\ell}}{\partial\boldsymbol{\Omega}_c^X} = \left(\hat{V}^{\pi_X}(\boldsymbol{X}_\ell) - \hat{V}^{T_X}(\boldsymbol{X}_\ell)\right)\boldsymbol{X}_\ell\,\boldsymbol{X}_\ell^T$, and t refers to the iteration update index.

Similarly, the squared approximation errors for the actor networks are defined by

$$\varepsilon_a^{E_\ell} = \frac{1}{2}\left(\hat{\boldsymbol{u}}_\ell^{\pi_E} - \hat{\boldsymbol{u}}_\ell^{T_E}\right)^2, \quad \varepsilon_a^{X_\ell} = \frac{1}{2}\left(\hat{\boldsymbol{u}}_\ell^{\pi_X} - \hat{\boldsymbol{u}}_\ell^{T_X}\right)^2,$$

where the target values $\hat{\boldsymbol{u}}_\ell^{T_E}$ and $\hat{\boldsymbol{u}}_\ell^{T_X}$ are calculated as follows

$$\hat{\boldsymbol{u}}_\ell^{T_E} = -\boldsymbol{P}^E\,\boldsymbol{\Omega}_c^E\boldsymbol{E}_\ell, \quad \hat{\boldsymbol{u}}_\ell^{T_X} = -\boldsymbol{P}^X\,\boldsymbol{\Omega}_c^X\boldsymbol{X}_\ell.$$

Using gradient descent, the actor network weights are adapted according to the following rule

$$\boldsymbol{\Omega}_a^{E(t+1)} = \boldsymbol{\Omega}_a^{E(t)} - \alpha_a^E \left[\frac{\partial \varepsilon_a^E}{\partial \boldsymbol{\Omega}_a^{E_\ell}}\right]^{(t)}$$
$$\boldsymbol{\Omega}_a^{X(t+1)} = \boldsymbol{\Omega}_a^{X(t)} - \alpha_a^X \left[\frac{\partial \varepsilon_a^X}{\partial \boldsymbol{\Omega}_a^{X_\ell}}\right]^{(t)}, \quad (8)$$

where $0 < \alpha_a^E, \alpha_a^X < 1$ are actor networks learning rates, $\frac{\partial \varepsilon_a^{E_\ell}}{\partial \boldsymbol{\Omega}_a^E} = \left(\hat{\boldsymbol{u}}_\ell^{\pi_E} - \hat{\boldsymbol{u}}_\ell^{T_E}\right) \boldsymbol{E}_\ell^T$, and $\frac{\partial \varepsilon_a^{X_\ell}}{\partial \boldsymbol{\Omega}_a^X} = \left(\hat{\boldsymbol{u}}_\ell^{\pi_X} - \hat{\boldsymbol{u}}_\ell^{T_X}\right) \boldsymbol{X}_\ell^T$.

A full schematic diagram of the adaptive critics process is shown in Figure 5 for the combined control problem in hand. The different actor and critic weights are updated simultaneously in real-time following the scope of Algorithm 1.

Algorithm 1: Online Value Iteration Process.

Input: Desired trajectory $\theta_\ell^{desired}$, weighting matrices $\boldsymbol{Q}^E, \boldsymbol{R}^E, \boldsymbol{Q}^X$, and $\boldsymbol{R}^X$, guiding search vectors $\boldsymbol{P}^E$ and $\boldsymbol{P}^X$, tracking error evaluation interval N.

Output: Policies $\boldsymbol{\pi}_E$, $\boldsymbol{\pi}_X$, and tracking error e_ℓ, for $\ell = 0, 1, \ldots$

```
begin
    ℓ = 0, t = 0 /* time and strategy indices */
    Initialize solving matrices S^{E{0}} and S^{X{0}}
     /* Positive definite */ and hence initial
     policies π_E^0 and π_X^0 and tracking errors
     interval e_{ℓ-1}, ..., e_{ℓ-N-1}
    Measure θ_ℓ, θ_{vℓ}, θ_{aℓ} and then calculate the
     errors e_ℓ, e_{ℓ-1}, e_{vℓ}, e_{vℓ-1}, e_{sℓ}, e_{sℓ-1}
    repeat/* Training/Search loop */
        Compute the different control signals
         u^{π_E(t)} and u^{π_X(t)} using (4)
        Measure θ_{ℓ+1}, θ_{vℓ+1}, θ_{aℓ+1} and then
         calculate the errors
         e_{ℓ+1}, e_ℓ, e_{vℓ+1}, e_{vℓ}, e_{sℓ+1}, e_{sℓ}
        Evaluate the solving value functions
         V^{π_E(t+1)}(E_ℓ) =
         C^E(E_ℓ, u_ℓ^{π_E(t)}) + V^{π_E(t)}(E_{ℓ+1})
         V^{π_X(t+1)}(X_ℓ) =
         C^X(X_ℓ, u_ℓ^{π_X(t)}) + V^{π_X(t)}(X_{ℓ+1}).
        Extract the improved control policies
         π_E^{t+1} = -P^E S^{E(t+1)}, π_X^{t+1} =
         -P^X S^{X(t+1)},
        ℓ ← ℓ + 1 /* Update real-time index */
        t ← t + 1/* Update policy index */
    until Satisfactory trajectory-tracking
     performance (i.e., acceptable tracking error).
```

D. Complexity of the Learning Mechanism

Algorithm 1 provides multiple merits concerning the complexity and scalability of the navigation or tracking problems. First, the learning process can be conditioned for any number of tracking signals simultaneously (i.e., for a large tracking or navigation process that contains many tracking objectives, that would be easy to implement compared to solving a high number of differential equations in an offline mode). Second, the tracking error signals could involve many dynamical forms (i.e., moving velocity, acceleration, moving average, etc) which were not possible or easy-to-implement before. Third, this scheme allows for coupled optimized dynamical processes (i.e., possible optimization of multiple coupled and interactive dynamical objectives for multiple problems without the need to independently solve each problem). Finally, the arrangement of Bellman equations enables simple and straightforward adaptive critics solution, since the control strategies hold linear forms.

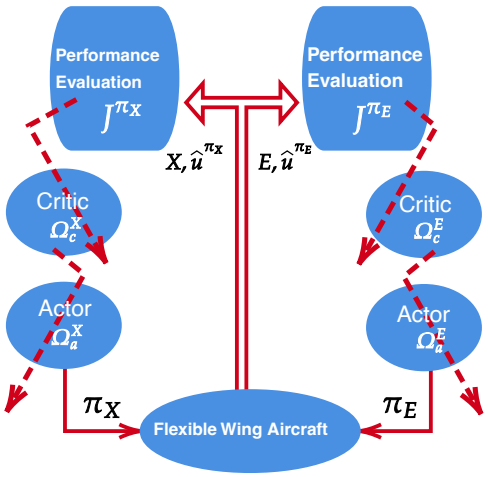


Fig. 5: Adaptive critics structure.

VI. Experimental Results

The online model-free adaptive learning mechanism is validated in a real-time environment using a two-servomotor pitch actuation system. The system generates servomotor pull actions dependent on the current-measured pitch position and the desired trajectory steering the wing. For the sake of experiments presented below, the trajectory reference is chosen to be sinusoidal (i.e., continuous pitch up/down commands).

A. Learning Environment

The learning parameters of the system are selected in order to reflect the physical constraints of the measured dynamical parameters and also encode the control design preferences that prioritize the desired system response. The weighting matrices capture the physical limitations of the dynamical parameters (i.e., the states $\theta_\ell, \theta_{v\ell}, \theta_{a\ell}, e_\ell$ and the actuation control signal u^F). They are chosen as $R^E = 10^{-7}$, $R^X = 10$,

$$Q^E = 10^{-4}\begin{bmatrix} 25 & 0 & 0 & 0 & 0 & 0 \\ 0 & 25 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.25 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.25 & 0 & 0 \\ 0 & 0 & 0 & 0 & 25 & 0 \\ 0 & 0 & 0 & 0 & 0 & 25 \end{bmatrix},$$

$$Q^X = 10^{-6}\begin{bmatrix} 25 & 0 & 0 \\ 0 & 25 & 0 \\ 0 & 0 & 0.0025 \end{bmatrix}.$$

Vectors P^E and P^X for the guided search policies π^E and π^X are selected so that $P^E = \begin{bmatrix} 200 & 50 & 10 & 5 & 10 & 5 \end{bmatrix}$ and $P^X = \begin{bmatrix} 10 & 10 & 5 \end{bmatrix}$. The vector P^E assigns more weight to minimizing the recently measured tracking errors e_ℓ and $e_{\ell-1}$, as opposed to the error velocity terms $e_{v\ell}$ and $e_{v\ell-1}$, to smooth down the nonlinearity transitions. Furthermore, the influences of the moving average tracking errors $e_{s\ell}$ and $e_{s\ell-1}$ on the overall performance are weighted similarly to those of the error velocities. The vector P^X reflects the gradual dynamic importance of the pitch attitude θ_ℓ and the pitch velocity $\theta_{v\ell}$ over the angular acceleration $\theta_{a\ell}$. The actor and critic learning rates are selected to be small enough in order to match the actor and critic adjustments in a smooth manner. They are set to $\alpha_c^E = 0.01$, $\alpha_c^X = 0.01$, $\alpha_a^E = 0.01$, and $\alpha_a^X = 0.05$. The desired navigation trajectory is an independent sinusoidal reference signal that takes the form $\theta_\ell^{desired} = 20\,\sin(0.132\,\pi\,\ell)$ deg.

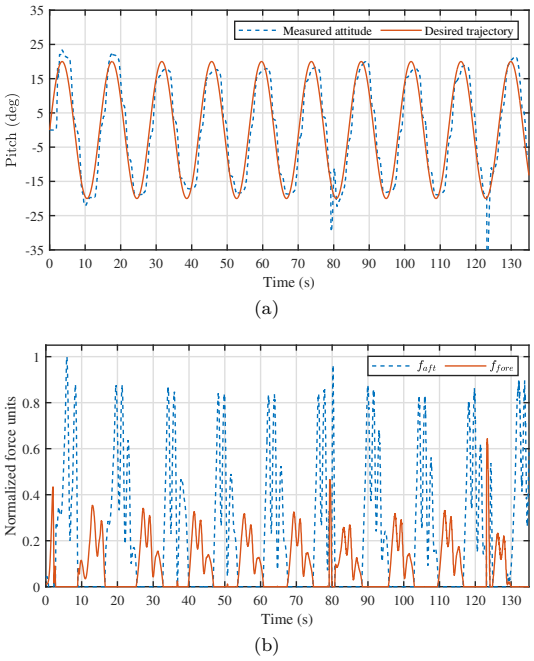


Fig. 6: Control performance during online learning process: (a) measured vs. desired pitch attitude, (b) acting forces on the wing's keel, f_{fore} and f_{aft}.

B. Test Scenarios

The adaptive learning controller is validated using three test scenarios. In the first scenario, the system is tested under nominal circumstances where no external disturbance is applied on the system. The dynamic performance of the proposed adaptive learning controller is shown in Figure 6. This experiment reveals that, the learning algorithm successfully converges to a stable control policy which prescribes correct servomotor pull-forces for the observed system state (i.e., pitch attitude). The absolute average tracking error over the test period is found to be 0.45 deg. It is worth noticing that the convergence behavior of the proposed control mechanism is not affected by the occasional sudden erroneous readings in the feedback measurements, as revealed at around time instants 80 s and 123 s in Figure 6a. Such measurement errors are due to some imperfections in the sensing units which could not be overcame by the adopted low-pass filter. The convergence of the adaptive learning approach is clear in Figure 7, where the actor and critic weights converge throughout the online learning process. The overall normalized forces generated by the servomotors are shown in Figure 6b. Finally, the convergence of the value iteration approach, as described in Section V, is highlighted in Figure 8. It is shown that, despite the sensor reading spikes, a general bounded non-decreasing convergence pattern is observed, as expected for the underlying value iteration process.

The second scenario is set up to demonstrate the ability of the controller to reject sudden external and forced disturbances. The test case is a repetition of the first, except that this time, a sudden disturbance was applied at around 120 s for about 10 s by violently jerking the control bar around the pitch and roll axes to stimulate the effects of wind gusts on the wing during the flight. The tracking performance of the proposed controller is demonstrated in Figure 9a. The controller is started with converged control gains from a previous learning episode. As such, minimal changes in the critic weights are witnessed, as shown by the critic weights evolution in Figure 10. During disturbance, the critic reacts appropriately to generate large abrupt motor torques to counteract the induced perturbations, as shown in Figure 9b. Once the dynamics are back to their nominal conditions, after that the disturbance subsided, the controller successfully resumes the tracking of the commanded pitch.

The goal of the third experiment is to show that once the controllers converge to some optimal strategies π_E^* and π_X^*, the actor's learning mechanism can be turned off without degrading the control performance. By doing so, the controller is made to operate with static control gains. For this test case, the adopted static control policies applied are those converged from a prior experiment. The controller's performance is illustrated in Figure 11. The absolute average tracking error over the test period is found to be 0.94 deg. This result emphasizes that the

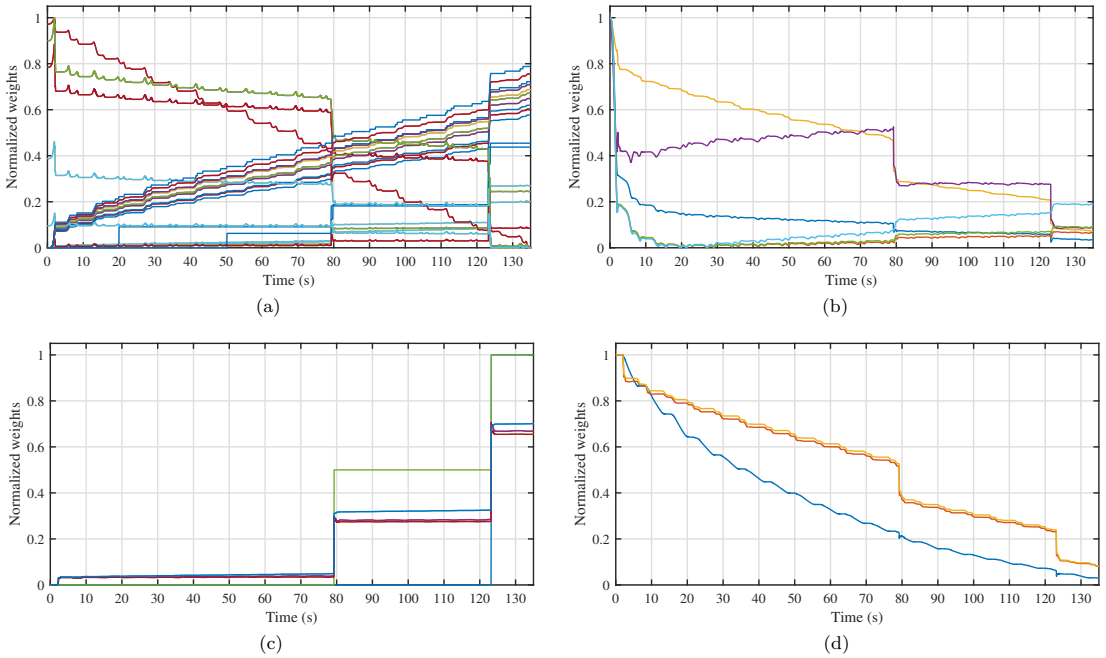


Fig. 7: Evolution of actor-critic weights during learning process: (a) tracking critic unit, (b) tracking actor unit, (c) stabilizing critic unit, (d) stabilizing actor unit.

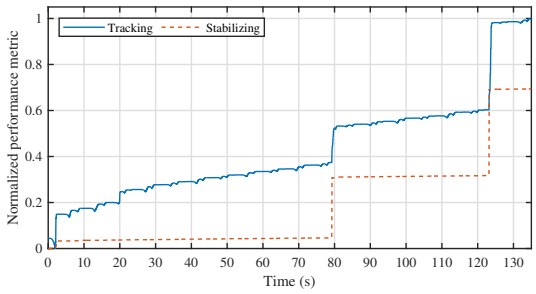


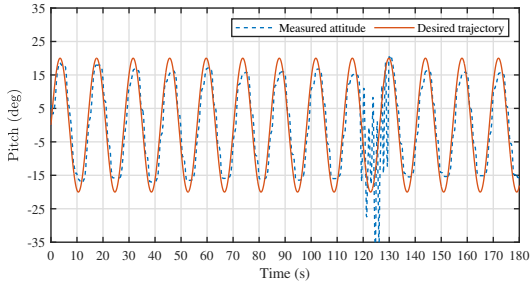
Fig. 8: Evolution of the policy evaluation metric.

converged strategies from static "actor-only" controllers previously derived from the adaptive learning mechanism remain valid during the trajectory tracking process. Notice how this experiment also witnessed two abrupt spikes in the sensor measurements.

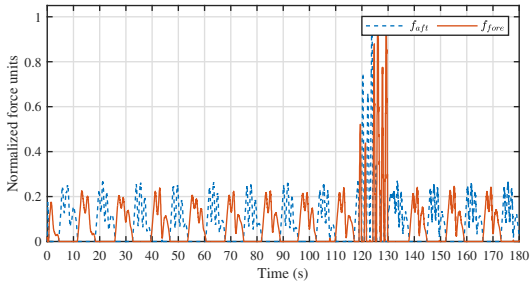
Unlike classical Q-leaning processes which employ multiple offline training episodes before settling on suitable control strategies, herein the proposed approach showed successful outcomes following a single real-time learning episode as shown in Figures 6a, 9a. Additionally, the learning process exhibits capability to handle large mechanical systems. This explains the superior features of the proposed approach to follow desired trajectory reaching bounded error when some converged polices are utilized to run the system as shown in Figure 11a. The maximum observed absolute average tracking error was confined to under 1 deg for such mechanical actuation system (without overlooking the sensory error biasing).

When experimenting with real systems (as opposed to simulation), it is typical to notice differences between the reference and the actual signals. The differences may be due to several reasons, including noise, delay caused by digital filters, and backlashes. In this particular work, the differences also come from the coupling between the roll and pitch motions. The two were assumed to be decoupled in the first-order linear approximation of the system. However, in a real system, such as the one we adopted, they are never completely decoupled. Nevertheless, it is clear from the figures that the lag between the reference and the actual signals is bounded within an acceptable margin.

Remark 3: As explained earlier, the stability of the value iteration solution was investigated and proved, provided that the system under consideration is stabilizable and convex objective cost function is adopted. Figures 6a, 9a, and 11a emphasize the bounded stability features and the ability of the online learning mechanism to retrieve stability under significant mechanical disturbances, noise, and false readings. These figures show that the tracking errors are bounded and non-increasing. This is reassured by examining the actuation input signals shown in Figures 6b, 9b, and 11b.

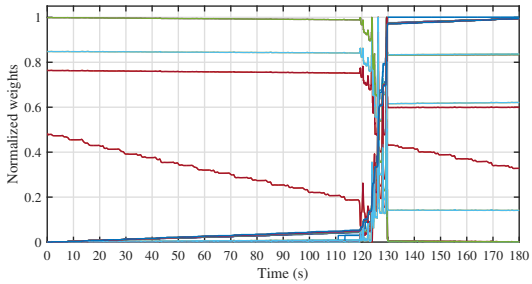


(a)

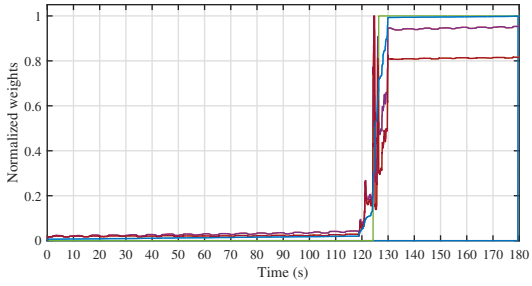


(b)

Fig. 9: Control performance while learning with a converged controller in the presence of mechanical disturbances: (a) measured vs. desired pitch attitude, (b) acting forces on the wing's keel, f_{fore} and f_{aft}.

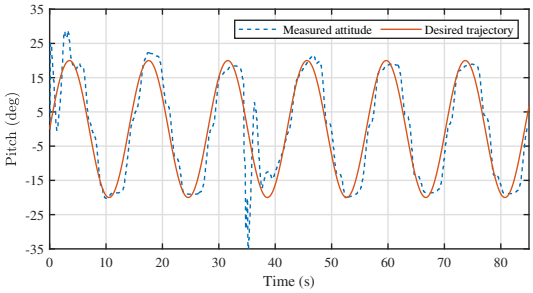


(a)

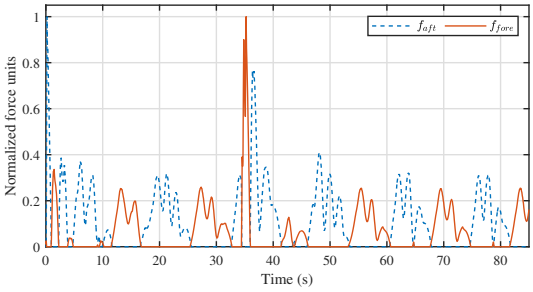


(b)

Fig. 10: Variations in critic weights in the face of mechanical disturbances: (a) tracking unit, (b) stabilizing unit.



(a)



(b)

Fig. 11: Control performance with previously learned static control policies: (a) measured vs. desired pitch attitude, (b) acting forces on the wing's keel, f_{fore} and f_{aft}.

VII. Conclusion

The challenging autonomous navigation of flexible-wing aircraft is solved by integrating analytical and computational solution platforms into an extensible experimental instrumentation and actuation incubator. The feedback loop receives the inertial measurements from standard measurement package Navio2 mounted on the wing system of an aircraft and decides, in real-time, on the best control strategies without acquiring any prior knowledge on the system's dynamical model. This enabled the integration of a powerful model-free control unit with a flexible and affordable measuring circuitry without over-complicating the sensory structures for such type of systems. The quality of the resultant systems depends mostly on the design of the model-free learning process rather than the precision of the sensors. This could be faced if a different model-based strategy is followed or even complicated augmented control structures are considered for this type of aircraft. This in turn, opens the door to generalize this approach for problems with similar interests. On another side, an online guided search mechanism based on a value iteration process is introduced where the learning process decides on the real-time control strategies based on the dynamic selection of the goals of the optimization problem. The experimental results coincided with the stated objectives, where the wing is subjected to severe

disturbances without destabilizing the system.

References

[1] Y. Ochi, "Modeling of flight dynamics and pilot's handling of a hang glider," in AIAA Modeling and Simulation Technologies Conference. American Institute of Aeronautics and Astronautics, 2017, pp. 1758–1776.

[2] M. V. Cook and E. A. Kilkenny, "An experimental investigation of the aerodynamics of the hang glider," in Proceedings of an International Conference on Aerodynamics, 1986.

[3] M. Abouheaf, W. Gueaieb, and F. Lewis, "Model-free gradient-based adaptive learning controller for an unmanned flexible wing aircraft," Robotics, vol. 7, no. 4, p. 66, 2018.

[4] F. Lewis, D. Vrabie, and V. Syrmos, Optimal Control, 3rd ed. New York, USA: John Wiley, 2012.

[5] E. Kilkenny, "Full scale wind tunnel tests on hang glider pilots," Cranfield Institute of Technology, College of Aeronautics, Department of Aerodynamics, Tech. Rep., 1984.

[6] E. A. Kilkenny, "An experimental study of the longitudinal aerodynamic and static stability characteristics of hang gliders," phdthesis, Cranfield University, Sep. 1986.

[7] D. Blake, "Modelling the aerodynamics, stability and control of the hang glider," Master's thesis, Centre for Aeronautics - Cranfield University, 1991.

[8] M. V. Cook, "The theory of the longitudinal static stability of the hang-glider," The Aeronautical Journal, vol. 98, no. 978, pp. 292–304, 1994.

[9] Y. Ochi, "Modeling of the longitudinal dynamics of a hang glider," in AIAA Modeling and Simulation Technologies Conference. American Institute of Aeronautics and Astronautics, 2015, pp. 1591–1608.

[10] J. Sweeting, "An experimental investigation of hang glider stability," Master's thesis, College of Aeronautics, Cranfield University, 1981.

[11] M. Cook, 'Flight Dynamics Principles. Butterworth-Heinemann, London, 2012.

[12] I. Kroo, Aerodynamics, Aeroelasticity and Stability of Hang Gliders. Stanford University, 1983.

[13] M. Spottiswoode, "A theoretical study of the lateral-directional dynamics, stability and control of the hang glider," mathesis, College of Aeronautics, Cranfield Institute of Technology, 2001.

[14] G. De Matteis, "Response of hang gliders to control," The Aeronautical Journal, vol. 94, no. 938, pp. 289–294, 1990.

[15] G. D. Matteis, "Dynamics of hang gliders," Journal of Guidance Control and Dynamics, vol. 14, no. 6, pp. 1145–1152, 1991.

[16] M. V. Cook and M. Spottiswoode, "Modelling the flight dynamics of the hang glider," The Aeronautical Journal, vol. 109, no. 1102, pp. I–XX, 2005.

[17] M. V. Cook, Ed., Flight Dynamics Principles: A Linear Systems Approach to Aircraft Stability and Control, 3rd ed., ser. Aerospace Engineering. Butterworth-Heinemann, 2013.

[18] M. Abouheaf, W. Gueaieb, and F. Lewis, "Model-free gradient-based adaptive learning controller for an unmanned flexible wing aircraft," Robotics, vol. 7, no. 4, p. 66, 2018.

[19] W. Gueaieb and M. S. Miah, "An intelligent mobile robot navigation technique using rfid technology," IEEE Transactions on Instrumentation and Measurement, vol. 57, no. 9, pp. 1908–1917, Sep. 2008.

[20] D. Esslinger, P. Rapp, S. Wiertz, H. Rendich, R. Marsden, O. Sawodny, and C. Tarín, "Accurate optoacoustic and inertial 3-d pose tracking of moving objects with particle filtering," IEEE Transactions on Instrumentation and Measurement, pp. 1–14, 2019.

[21] N. El-Sheimy, K. . Chiang, and A. Noureldin, "The utilization of artificial neural networks for multisensor system integration in navigation and positioning instruments," IEEE Transactions on Instrumentation and Measurement, vol. 55, no. 5, pp. 1606–1615, Oct 2006.

[22] X. Tao, J. Yi, Z. Pu, and T. Xiong, "State-estimator-integrated robust adaptive tracking control for flexible air-breathing hypersonic vehicle with noisy measurements," IEEE Transactions on Instrumentation and Measurement, pp. 1–15, 2019.

[23] H. Deng, Q. Fu, Q. Quan, K. Yang, and K. Cai, "Indoor multi-camera based testbed for 3d tracking and control of uavs," IEEE Transactions on Instrumentation and Measurement, pp. 1–1, 2019.

[24] K. Das Sharma, A. Chatterjee, and A. Rakshit, "A pso–lyapunov hybrid stable adaptive fuzzy tracking control approach for vision-based robot navigation," IEEE Transactions on Instrumentation and Measurement, vol. 61, no. 7, pp. 1908–1914, July 2012.

[25] I. A. R. Ashokaraj, P. M. G. Silson, A. Tsourdos, and B. A. White, "Robust sensor-based navigation for mobile robots," IEEE Transactions on Instrumentation and Measurement, vol. 58, no. 3, pp. 551–556, March 2009.

[26] R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction, 2nd ed., ser. Second. Massachusetts: MIT Press, 1998.

[27] P. Werbos, "Beyond regression: New tools for prediction and analysis in the behavior sciences," Ph.D. dissertation, Harvard University, 1974.

[28] P. J. Webros, "A menu of designs for reinforcement learning over time," in Neural Networks for Control, W. T. Miller, III, R. S. Sutton, and P. J. Werbos, Eds. Cambridge, MA, USA: MIT Press, 1990, pp. 67–95.

[29] M. Abouheaf and F. Lewis, "Approximate dynamic programming solutions of multi-agent graphical games using actor-critic network structures," in International Joint Conference on Neural Networks (IJCNN), Aug. 2013, pp. 1–8.

[30] M. Abouheaf, F. Lewis, K. Vamvoudakis, S. Haesaert, and R. Babuska, "Multi-agent discrete-time graphical games and reinforcement learning solutions," Automatica, vol. 50, no. 12, pp. 3038–3053, 2014.

[31] M. I. Abouheaf and F. L. Lewis, "Multi-agent differential graphical games: Nash online adaptive learning solutions," in 52nd IEEE Conference on Decision and Control, Dec 2013, pp. 5803–5809.

[32] R. Bellman, Dynamic Programming. Princeton University Press, 1957.

[33] A. Bryson, "Optimal control-1950 to 1985," IEEE Control Systems, vol. 16, no. 3, pp. 26–33, 1996.

[34] M. Abouheaf, F. Lewis, M. Mahmoud, and D. Mikulski, "Discrete-time dynamic graphical games: Model-free reinforcement learning solution," Control Theory and Technology, vol. 13, no. 1, pp. 55–69, 2015.

[35] M. Abouheaf and W. Gueaieb, "Multi-agent reinforcement learning approach based on reduced value function approximations," in 2017 IEEE International Symposium on Robotics and Intelligent Sensors (IRIS), Oct 2017, pp. 111–116.

[36] T. Başar and G. J. Olsder, Dynamic Non-cooperative Game Theory, 2nd ed., ser. Classics in Applied Mathematics:. SIAM: Philadelphia, 1999.

[37] B. Widrow, N. K. Gupta, and S. Maitra, "Punish/reward: Learning with a critic in adaptive threshold systems," IEEE Transactions on Systems, Man, and Cybernetics, vol. SMC-3, no. 5, pp. 455–465, 1973.

[38] P. Werbos, "Neural networks for control and system identification," in 28th Conference on Decision and Control, Dec. 1989, pp. 260–265.

[39] P. J. Webros, "Neurocontrol and supervised learning: An overview and evaluation," in Handbook of Intelligent Control: Neural, Fuzzy, and Adaptive Approaches, D. A. White and D. A. Sofge, Eds. Van Nostrand Reinhold, Jun. 1992, pp. 65–89.

[40] M. Abouheaf and M. Mahmoud, "Policy iteration and coupled riccati solutions for dynamic graphical games," International Journal of Digital Signals and Smart Systems, vol. 1, no. 2, pp. 143–162, 2017.

[41] M. I. Abouheaf, F. L. Lewis, and M. S. Mahmoud, "Differential graphical games: Policy iteration solutions and coupled riccati formulation," in 2014 European Control Conference (ECC), June 2014, pp. 1594–1599.

[42] D. Bertsekas and J. Tsitsiklis, Neuro-Dynamic Programming, 1st ed. Massachusetts: Athena Scientific, 1996.

[43] L. Busoniu, R. Babuska, and B. D. Schutter, "A comprehensive survey of multi-agent reinforcement learning," IEEE

Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), vol. 38, no. 2, pp. 156–172, 2008.
[44] M. Gomez Plaza, T. Martinez-Marin, S. Sanchez Prieto, and D. Meziat Luna, "Integration of cell-mapping and reinforcement-learning techniques for motion planning of car-like robots," IEEE Transactions on Instrumentation and Measurement, vol. 58, no. 9, pp. 3094–3103, Sep. 2009.
[45] M. Abouheaf, N. Mailhot, and W. Gueaieb, "An online reinforcement learning wing-tracking mechanism for flexible wing aircraft," in 2019 IEEE International Symposium on Robotic and Sensors Environments (ROSE), June 2019, pp. 1–7.
[46] Emlid Limited. Emlid official open source documentation. [Online]. Available: https://github.com/emlid/emlid-docs
[47] Raspberry Pi Foundation. Raspberry Pi official open source documentation. [Online]. Available: https://github.com/raspberrypi/documentation
[48] ArduPilot Development Team and Community. ArduPilot open source autopilot. [Online]. Available: http://ardupilot.org/
[49] BYU MAGICC Lab, Provo, Utah. ROS Flight. [Online]. Available: https://rosflight.org/
[50] TDK Corp. InvenSense. TDK InvenSense MPU-9250 sensor documentation. [Online]. Available: https//www.invensense.com/products/motion-tracking-9-axis/mpu-9250/
[51] STMicroelectronics N.V. STM LSM9DS1 sensor documentation. [Online]. Available: https://www.st.com/en/mems-and-sensors/lsm9ds1.html
[52] Robotis Co. Dynamixel SDK. [Online]. Available: http://emanual.robotis.com/docs/en/software/dynamixel/dynamixel_sdk/overview/
[53] M. Abouheaf and F. Lewis, Dynamic Graphical Games: Online Adaptive Learning Solutions Using Approximate Dynamic Programming. World Scientific, 2014, ch. Chapter 1, pp. 1–48.
[54] T. Landelius and H. Knutsson, "Greedy adaptive critics for lqr problems: Convergence proofs," Neural Computation - NECO, 01 1996.

A.3 uWSC Aircraft Simulator: A Gazebo-Based Model for Uncrewed Weight-Shift Control Aircraft Flight Simulation

This is the accepted version of the paper. Published paper DOI: 10.1109/ROSE60297.2023.10410672.©2023 IEEE. Reprinted, with permission, from uWSC Aircraft Simulator: A Gazebo-based model for uncrewed weight-shift control aircraft flight simulation, IEEE International Symposium on Robotics and Sensors Environments, 11/2023.

uWSC Aircraft Simulator: A Gazebo-based model for uncrewed weight-shift control aircraft flight simulation

Nathaniel Mailhot
Department of Mechanical Engineering
University of Ottawa
Ottawa, ON, Country
nmailhot@uottawa.ca

Teresa de Jesus Krings
Department of Mechanical and Aerospace Engineering
Carleton University
Ottawa, ON, Canada
teresadejesuskrings@cmail.carleton.ca

Gilmar Tuta Navajas
Romaeris Corporation
Ottawa, ON, Canada
gilmar.tuta@romaeris.com

Boyan Zhou
Department of Mechanical and Aerospace Engineering
Carleton University
Ottawa, ON, Canada
boyanzhou@cmail.carleton.ca

Davide Spinello
Department of Mechanical Engineering
University of Ottawa
Ottawa, ON, Canada
dspinell@uottawa.ca

***Abstract*—Weight-shift control (WSC) microlights are a class of aircraft that maneuver by manipulating a mass attached to deforming flexible wings, which are favoured by many recreational pilots due to their simple mechanical structure, superior low-speed maneuverability, positive aerodynamic stability, and capability to operate on rugged terrains with minimal infrastructure requirements for take-off and landing. However, the dependency on human-powered control restricts their effective payload and mission scope. Here, we present an uncrewed weight-shift control (uWSC) aircraft simulator model developed within the open-source robotics simulator Gazebo. The aim of the model is to spur further study of weight-shift aircraft as a new class of uncrewed aerial systems. The model's physical properties are based upon an uncrewed electrically powered hang-glider prototype. The longitudinal and lateral positive static aerodynamic stability of the flexible hang-glider wing is approximated by a rigid wing consisting of multiple joined sections. The performance of the uWSC aircraft model is benchmarked against data captured from real weight-shift aircraft flight tests. Results indicate that the simulator model provides a suitable approximation of real weight-shift aircraft flight. The simulator's capabilities in emulating weight-shift control mechanisms in flight is indicative of its utility in bridging higher-level flight controller design with lower-level weight-shift control dynamics, opening new directions towards the autonomous and semi-autonomous operations of uWSC by allowing quick testing for design and prototype innovation and improvement.**

***Index Terms*—uncrewed aircraft, weight-shift control, flight simulator, flight data**

I. Introduction

Weight-shift control (WSC) microlights are a class of aircraft that maneuver by manipulating a mass attached to deforming flexible wings [1]. Many recreational pilots favour WSC aircraft due to their simplistic mechanical structure, superior low-speed maneuverability, positive aerodynamic stability, and capability to operate on rugged terrains with minimal infrastructure requirements for take-off and landing [2], [3]. However, the need for human-powered control restricts their effective payload and mission scope. As a result, WSC aircraft are primarily used for recreation by enthusiast pilots, with limited industrial applications. This niche market has resulted in a lack of significant engineering development focus on these aircraft. New prototype aircraft designs developed by Romaeris Corporation integrate robotic piloting systems to enable uncrewed weight-shift control (uWSC), unlocking their use as remotely piloted aircraft systems [4]–[6]. The uWSC aircraft simulator model introduced in this work approximates the dynamics of the Romaeris prototype aircraft that it is based upon. Figure 1 presents a capture from simulation of the model performing a simulated flight. The primary intent of this open-source contribution is to provide a critical simulation tool to support further study of uWSC aircraft control systems.

The development of automatic flight control for weight-shift aircraft is nascent. While there is some research on modelling the dynamics of this class of systems, it remains limited and specialized. Accurate dynamic models are challenging to develop as the aircraft have highly nonlinear dynamics due to the flexible wing's airfoil deformation, as well as two-body weight-shift relationship that is inherent to their flight control [7]. Flexible wing weight-shift control aircraft trace their routes to engineering designs developed by Rogallo, with the original intended application of recovering space vehicles after atmospheric entry [8]. As they became adopted for sport flying, all following design evolution of hang-gliders was done on a trial and error basis, absent of typical scientific

This work was supported by Romaeris Corporation and Mitacs under the Accelerate fellowship project *Artificial Intelligence (AI) Powered Adaptive Flight Controller for Novel Unmanned Aerial Vehicle (UAV) with Commercial and Humanitarian Applications.*

Fig. 1. Capture from Gazebo simulator of the uWSC aircraft model performing a starboard side banking turn. The hung mass under the wing is responsible for attitude adjustments via weight-shifting control.

rigour, resulting in poorly understood stability properties and high accident rates [9], [10]. The earliest comprehensive aerodynamic models were developed by using data from flying rigs attached to ground vehicles to produce conditions for experiments, resulting in aeroelastic and aerodynamic models for the wing that identified their stability properties [11], [12]. In the early 1980's, Kilkenny embarked upon her program to ascertain the aerodynamic properties for hang-glider wings with experimental investigations using full-scale wind-tunnel experiments, resulting in an experimentally validated model for the longitudinal dynamics of the hang-glider [10], [13]–[15]. Near the same time, preliminary longitudinal and lateral dynamic models of hang-glider wings were developed by de Matteis et al. that assume a rigid wing while neglecting the velocity dependence of the deforming airfoil shape [16], [17]. These models were experimentally confirmed recently with scaled wind-tunnel test performed by Khaddage [18]. Cook advanced the field by developing theory for the longitudinal static stability of hang-gliders [19], [20]. These models were further developed to describe nonlinear effects due to variable wing camber, wing twist, and the impact of the weight-shift effect of the suspended mass upon flight control and stability [21], [22], culminating in a sophisticated flight dynamics model of the hang-glider [23], [24].

The mathematical model developed by Cook et al. consists of a system of nonlinear dynamic equations that describe all kinetic phenomena of the hang-glider. The modelled dynamic properties include the force and moment balances for inertial, gravitational, and aerodynamic lift and drag, as well as the trim equilibrium conditions. Small aerodynamic and pilot-induced disturbances in the motion variables are represented by a set of stability and control derivatives by techniques that match those found in classical flight control textbook treatments [25]. The most pertinent aspect presented in [24] is the complicated treatment of an aerodynamic model for the deforming flexible wing. The model replicates the main aerodynamic features of the deforming airfoils. The modelled flexibility features map deformation from wing luff, camber, and twist to corresponding aerodynamic effects on lift, drag, trim, and stability and control derivatives, all of which inform the overall stability properties for the aircraft in response to control inputs of the pilot. It is limited in that it is quasi-static about equilibrium trim conditions and does not account for non-steady aerodynamic conditions or attached hanging mass's weight-shift. Ochi built further upon the model for use in dynamics simulation work by developing the equations of motion for the two-body system where the hanging mass, i.e. the hang-glider pilot, is considered and held in place relative to the wing by a holonomic constraint that represents the hang-strap [26], [27]. The equations of motion for the two-body system are analytically obtained, and exploited to develop a candidate system model for the human pilot control input. Notably, Ochi concludes that although the analytically derived nonlinear dynamic model is unstable, the addition of the pilot control system in simulation stabilizes the aircraft. Most recently, Abouheaf et al. validated the model developed by Ochi, and designed data-driven adaptive control approaches based on reinforcement learning that can pilot Ochi's weight-shift aircraft model [28]–[33].

Simulation tools present a valuable avenue for the testing and refinement of weight-shift aircraft, building on the foundation established by past modelling efforts. The combination of virtual dynamics models acting within simulated environment enables rapid validation of prototype systems without incurring real experiment cost [34], [35]. Gazebo is a widely used open-source simulation tool among roboticist for its compatibility with various robot systems and environments [36]. Common applications of Gazebo include autonomous wheeled robots, stationary manipulators, and biomimetic locomotive systems. [37]–[42]. Recent research efforts have equipped Gazebo simulation with rudimentary models for fluid dynamics phenomena to enable the study of water-borne and air-borne drones. Hydrodynamic and hydrostatic effects have been implemented for the study of uncrewed underwater vehicles in [43], while the simulation of flying rotor-craft, fixed wing airplanes, and airships are enabled by simplistic airfoil and buoyancy models [44]–[49]. Pertinent to these efforts is the implementation of hardware-in-the-loop and software-in-the-loop architectures that allow for real-time experimental flight controllers to be validated within simulation environments [50]–[52]. The significant efforts undertaken in this area provide valuable technical insights that are leveraged to facilitate a simulation methodology to support the development of weight-shift aircraft flight controls.

To the best of the authors' knowledge, the uWSC aircraft model presented in this paper represents the first open-source simulation model of a weight-shift control aircraft. The remainder of the paper is organized as follows: The dynamics model for the uWSC aircraft simulator model based on the Romaeris prototype design is presented in Section II. In Section III a set of benchmarks is defined to quantify flight performance of the simulated WSC flight against a set of data collected from real WSC aircraft flight test. Conclusions and future work are summarized in Section IV.

II. Weight-shift aircraft Gazebo model

The uWSC aircraft simulator model developed with the Gazebo simulator is based on the Romaeris aircraft prototype in Figure 2. The aircraft prototype consists of two primary bodies: the wing structure which includes the landing gear, and the weight-shifting mass that hangs beneath the wing which includes the batteries, propellers and payload. The two bodies are attached by a two-axis revolute joint on the wing keel named the hang-block, allowing the shifting of hung mass to produce attitude control by weight-shift adjustments. The software-in-the-loop feature of ArduPlane is employed to simulate automatic flight control [51], [53]. Attitude adjustment of the wing via fuselage weight-shifting is induced by direct control of the hang-block joint angles.

Fig. 2. 3D computer-aided design model of the Romaeris uncrewed weight-shift aircraft prototype.

The Gazebo model approximates the prototype aircraft's mechanical subsystems. Table I presents the Gazebo model's primary components' mass, inertia, and relative center of mass properties that were calculated from a 3D computer-aided design model. The layout of the primary components are visualized with RVIZ in Figure 3, and the corresponding tree diagram visualization of the model's link structure is presented by Figure 4 [54].

The prototype aircraft is equipped with a Wills Wing Falcon 3 170 [55] that is rated for a maximum velocity of $77\,\mathrm{km/h}$ and a maximum takeoff load of $130\,\mathrm{kg}$. Expert weight-shift pilots and aerodynamic modelling references are consulted to derive a reasonable approach to approximate the wing's aerodynamic stability properties as a rigid component within Gazebo [2], [18], [24], [49]. The approach taken to model correct longitudinal dynamic stability features of the flexible wing is summarized as follows:

- Wing tips have smaller angle of attack (AoA), while the wing root has a larger AoA.
- Wing tips generate lift as well as pitch down moment.
- Wing root generates lift as well as pitch up moment.
- When the aircraft is pitching up, the wing root tends to stall. The pitch up moment from the wing root will decrease while the pitch down moment from the wing tip will increase. Then the aircraft will tend to pitch down.

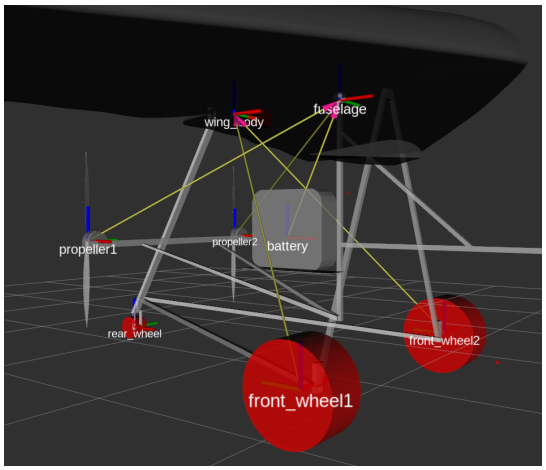


Fig. 3. Gazebo model visualized with primary components labelled.

- When the aircraft is pitching down, the wing tip will have negative AoA angle and the pitch down moment from the lift will decrease. At this point, the wing root will have smaller but positive AoA so the pitch up moment will increase, then the aircraft will tend to pitch up.

The approach taken to model correct lateral dynamic stability features of the flexible wing is summarized as follows:

- Wing tips have a dihedral angle (upward inclination of surface in lateral axis), while wing roots have anhedral angle (downward inclination of surface in lateral axis).
- Wing tips generate lift as well as an inward rolling moment.
- Wing root generates lift as well as an outward rolling moment.
- When tuning the lateral aerodynamic performance, more dihedral angle on wingtips and less anhedral angle on wing roots will produce a better lateral stability range. Doing the inverse will trade-off stability range for increased performance in flight, namely higher flight speeds with less energy consumption.
- When a roll occurs to one direction, relative deflection of the wing tips occurs, with the lower tip creating a higher lift than the opposite tip, inducing a stabilizing roll moment. That is, as the port side tip moves downwards, the starboard side tip will increase in lift, inducing a restoring moment towards the trimmed state.

The Gazebo wing model is divided into four sections to achieve the desired aerodynamic stability properties. Each half of the wing consist of two sections, named the wing tip and wing root. Aerodynamic properties for the sections are confirmed through analysis using XFLR5 [56], where the prototype's wing is modelled as an MH-82 hang-glider airfoil. The wing roots are attached by a revolute joint to the parent keel link, with a very large stiffness and friction parameter to limit most motion while allowing very small deflections,

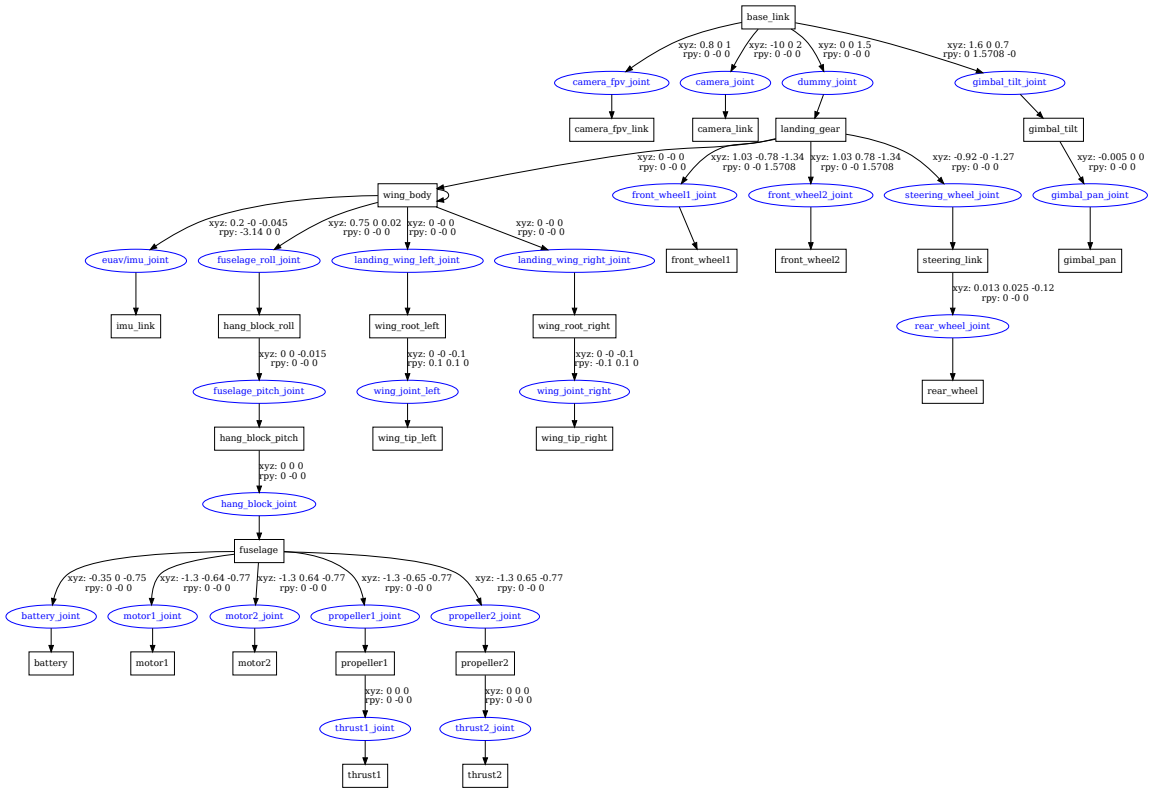


Fig. 4. Tree diagram representation of the uWSC aircraft model as defined by the unified robotics description format.

such that the simulator's physics engine applies independent aerodynamic forces to each section. The same process is repeated to attach the wing tips to the wing roots. The wing roots possess a larger effective AoA compared to the wing tips. Table II presents the aerodynamic parameters for the wing root and wing tip sections of the uWSC aircraft model.

Aerodynamic modelling of the propellers is more straightforward as matching thrust levels of the prototype aircraft design was the only requirement. Table III presents the aerodynamic properties for the propellers of the uWSC aircraft model.

III. Flight Benchmarking Results

The flight performance of the uWSC aircraft model is benchmarked against real-time flight data from thirteen real-world human-piloted weight-shift aircraft missions, provided by Romaeris Corporation. The flights were performed by a Cosmos Phase 3 ultralight fuselage equipped with a La Mouette IPSOS 19 flexible wing, [57], [58], equipped with a Navio2 flight controller running ArduPlane purely for data collection purposes [59]. The wing is rated for a maximum velocity of $80\,\mathrm{km/h}$, and a total flying weight of $350\,\mathrm{kg}$. Thirteen simulated missions that replicate the real flight trajectories using the uWSC aircraft model are conducted to enable the comparison. The model's quality is validated by comparing simulated flights to real flight data.

The aggregate statistics of all thirteen pairs of flight trajectories are calculated for a set of key flight dynamics parameters; this consist of the measurements for airspeed, wing roll angle, wing pitch angle, wing AoA, side-slip angle, turn rate, and climb rate. Additionally, statistics are calculated for subsets of the flight trajectories along four phases of flight, defined as follows:

- Take-off, representing the initial phase of flight where the aircraft accelerates from the ground to a pre-determined altitude, considered from the start of motion along the runway to the point the aircraft reaches $10\,\mathrm{m}$ above ground level and $13\,\mathrm{m/s}$.
- Landing, representing the final phase of flight where the aircraft descends and touches down on the runway, considered from $10\,\mathrm{m}$ above ground level and coming to a stop within $25\,\mathrm{s}$.
- Level, representing stable flight where the aircraft maintains a relatively constant attitude, heading, and speed,

TABLE I
MASS, INERTIA TENSOR, AND RELATIVE CENTER OF MASS LOCATION FOR PRIMARY COMPONENTS OF THE UWSC AIRCRAFT GAZEBO MODEL

Link name	**Mass** $[kg]$	**Inertia** $[kg \cdot m^2]$						**Relative position** $[m]$		
		Ixx	Ixy	Ixz	Iyy	Iyz	Izz	X	Y	Z
Wing root, left	8.0	102.6	0.09	0	9.56	-3.1e-3	112.14	-0.5	0	-0.75
Wing root, right	8.0	102.6	0.09	0	9.56	-3.1e-3	112.14	0.5	0	-0.75
Wing tip, left	3.0	41.1	0.03	0	3.82	-1.2e-3	44.96	-0.5	0	0
Wing tip, right	3.0	41.1	0.03	0	3.82	-1.2e-3	44.96	0.5	0	0
Landing gear	18.8	2.05	0.03	-0.08	4.36	-1.6e-2	3.80	0.5	-0.03	-1.2
Fuselage	14.6	0.56	0	-0.16	2.68	0	2.67	0	0	-1
Battery	17.1	0.32	0	0	0.6	0	0.35	0	0	-0.5
Motor	1.74	0.01	0	0	3.0e-3	0	3.0e-3	0	0	0
Hang-block	0.64	0	0	0	0	0	0	0	0	0
Propeller	0.23	0	0	0	1.68e-2	0	1.68e-2	0	0	0

TABLE II
AERODYNAMIC PARAMETERS PERTAINING TO WING ROOT AND WING TIP SECTIONS OF THE UWSC AIRCRAFT GAZEBO MODEL

Parameter	**Wing Root**	**Wing Tip**	**Rationale**	**Notes**
α_0	$-11.9°$	$-20.1°$	C_L considered as AoA at zero lift.	Selected from wing performance specification [55].
$C_{L\alpha}$	2.1	2.1	C_L slope before stall.	Estimated from the shape of the airfoil with XFLR5.
$C_{D\alpha}$	0.263	0.63	C_D slope before stall.	Estimated from the shape of the airfoil with XFLR5.
α_{stall}	$28.7°$	$28.7°$	AoA at stall point.	Estimated based on aircraft wing performance spec.
$C_{L\alpha_{stall}}$	-3.88	-2.88	C_L slope after stall.	Estimated based on aircraft wing performance spec.
$C_{D\alpha_{stall}}$	-0.43	-0.43	C_D slope after stall.	Estimated from the shape of the airfoil with XFLR5.
S	4.75 m^2	$2.92m^2$	Area of a single wing section.	Estimated from the wing CAD model.
C_p	$0.35 \pm 0.05\,\mathrm{m}$	$0.035 \pm 0.02\,\mathrm{m}$	Centre of aerodynamic pressure.	Dependent on wing incidence angle.

TABLE III
AERODYNAMIC PARAMETERS PERTAINING TO PROPELLER AIRFOIL OF THE UWSC AIRCRAFT GAZEBO MODEL

Parameter	**Value**	**Rationale**	**Notes**
C_p	$0.09\,\mathrm{m}$	A half of the total propeller airfoil length.	Constant C_p is assumed hence its location is about a quarter length of the airfoil since that is the assumed profile for the propeller blades, from the airfoil's leading edge.
Area	$0.04\,\mathrm{m}^2$	Area of a single propeller section.	From prototype CAD model for surface area of the propeller airfoil.
Multiplier	595	Produces simulated thrust force value verified to match experimental thrust test of 150 N per motor at maximum rotation.	Gazebo parameter which maps rotation to a force in axis incident to object rotation.

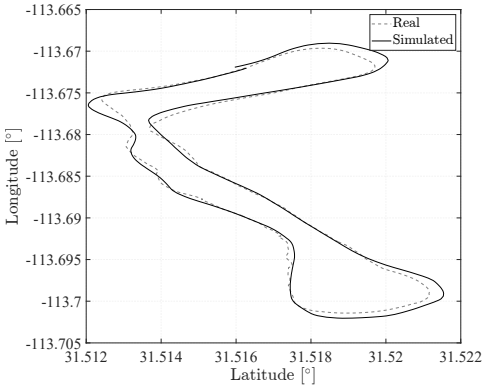
limited to altitude change of no more than $5\,\mathrm{m}$ and a heading change of no more than $40°$.
- Turn, representing a maneuver where the aircraft changes its heading to change course, considered as a turn rate of $3\,°/\mathrm{s}$ and a heading change of $90°$ at minimum.

The total flight trajectory data and the defined flight phase section data are interpreted to make direct quantitative comparisons between the real and simulated WSC flights. Note that although the aircraft modelled in the simulation differs from the real aircraft, the purpose is to assess the simulation's ability to match the characteristics of real WSC flights

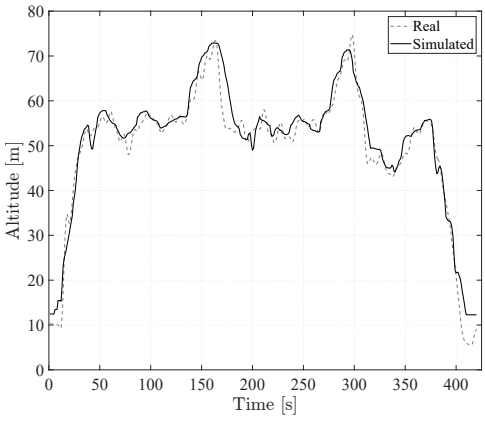
Table IV compares the statistics of key flight dynamics parameters between the real flights and their corresponding simulated flights. Several observations are drawn from the statistical results. Simulated airspeeds are generally lower than real ones, except during take-off where they align closely. Notable differences emerge in roll during turns and pitch values in general flights, with real data showing higher values than simulated. The large difference in the wing pitch mean is attributed to the different trim conditions for each aircraft. The mean AoA during take-off in real flights is considerably lower than simulations, due to the standard pilot practice of performing a rapid climb out with a large nose-up maneuver after reaching sufficient speed, where as the programmed take-off is gradual by comparison. Side-slip values during landing indicate a slight drift in real flights absent in simulations. Additionally, real-world flight data often exhibits higher variability, suggesting that real flights are influenced by more unpredictable factors than the simulations capture, such as wind conditions, turbulence, and unregulated control actions of the human pilot. This indicates a need for refinements in the simulation model to better represent actual flight conditions. However, while there are areas of discrepancy, notably in parameters like airspeed and pitch, in several instances such as airspeed during landing and side-slip in level flight, the simulated values closely align with the real-world data. The AoA values in general and level flight phases also reflect a remarkable similarity between real and simulated flights. This showcases the effectiveness of the current simulation model in capturing key aspects of real flight dynamics for WSC aircraft.

TABLE IV
COMPARISON OF KEY FLIGHT DYNAMICS MEAN AND STANDARD DEVIATION FOR THE REAL AND SIMULATED FLIGHT TRAJECTORIES OVER ALL THIRTEEN PROVIDED SAMPLES.

	Aggregate				**Take-off**				**Landing**				**Level**				**Turn**			
	Real		**Simulated**		**Real**		**Simulated**		**Real**		**Simulated**		**Real**		**Simulated**		**Real**		**Simulated**	
Parameters	Mean	Sd.	Mean	Sd.	Mean	Sd.	Mean	Sd.	Mean	Sd.	Mean	Sd.	Mean	Sd.	Mean	Sd.	Mean	Sd.	Mean	Sd.
Airspeed [m/s]	17.4	2.4	13.5	2.1	11.4	7.9	11.3	2.8	14.8	4.4	8.6	3.2	17.5	1.5	13.1	2.0	18.0	1.7	14.4	1.8
Roll [°]	1.2	9.3	0.3	5.2	0.8	3.9	0.0	0.5	1.3	4.6	0.1	3.2	0.5	7.1	0.1	3.4	4.9	14.0	2.1	8.6
Pitch [°]	16.2	4.7	0.2	3.4	11.9	9.8	5.7	2.4	10.4	4.9	1.4	3.3	17.0	2.9	-0.2	1.9	16.0	3.8	1.6	3.3
AoA [°]	16.0	3.5	14.7	2.0	8.7	7.4	16.8	1.8	11.2	5.9	18.6	8.1	16.7	2.7	14.4	1.1	16.0	2.7	15.3	2.0
Side-slip [°]	-0.6	5.2	-0.5	9.1	1.3	5.8	0.7	1.6	-3.4	3.9	-0.5	16.4	0.1	4.3	-0.6	7.9	-0.7	5.6	0.6	12.3
Turn rate [°/s]	0.3	4.1	0.2	2.9	-0.2	0.9	0.0	0.0	0.3	1.0	0.8	3.1	-0.1	2.6	0.1	1.6	1.6	6.1	1.2	4.6
Climb rate [m/s]	0.0	1.4	0.0	0.9	0.5	2.4	0.8	0.9	-0.3	1.4	-0.6	0.9	0.0	0.8	0.0	0.5	0.0	1.2	0.0	0.9



(a)



(b)

Fig. 5. (a) Projected ground track and (b) altitude over time for a single mission, comparing simulated and real flight data.

The performance of the simulated aircraft is further examined by a point to point comparison for a mission track. Figure 5 presents the two dimensional latitude-longitude plane projection of the trajectory as well as the altitude over time for a single mission sample. It is observed that the simulated aircraft presents less fluctuations in altitude in contrast to the real aircraft counterpart.

Figure 6 presents a qualitative comparison of the simulated aircraft's ability to match the three-dimensional trajectories of the real aircraft along the four flight phase subcategories. Due to automated flight and simplified environmental conditions in the simulation, it's evident that simulated flights exhibit more consistent and quicker take-off and landing phases compared to the varied and lengthier real-life counterparts. The landing phases for the real flight data presents offsets in the altitude level that suggest altimeter calibration issues. The simulated flights are observed to present accurate recreations of the real flight trajectories in the level and turn phases where the simulated flight controller aims to track the trajectory prescribed by the real flight data, in line with expectations suggested by the calculated statistics. The turns of the simulated aircraft are more gradual due to restrictions imposed on the attitude adjustments of the automatic flight control. Overall, the results suggest that the simulated aircraft provides a good approximation of real-world WSC aircraft flight.

IV. CONCLUSION

The uWSC aircraft simulator model offers a means to simulate the dynamics of weight-shift aircraft through Gazebo. The model's fidelity was validated by comparison from real-time data of thirteen weight-shift aircraft missions by Romaeris Corporation. While certain flight dynamic parameters, notably airspeeds and wing pitch, showcased differences between the simulated and real flights, several other parameters like the AoA values and side-slip in level flight demonstrate a good resemblance. The simulator's capabilities in emulating weight-shift control mechanisms in flight is indicative of its utility in bridging higher-level flight controller design with lower-level WSC dynamics. The overall consistency observed in various flight parameters affirms the model's use as a tool for advancing research in uncrewed weight-shift aircraft systems.

In the future, the uWSC aircraft model will be contributed to the open-source ArduPilot Gazebo repository once its refinement is complete. Flight data collected from the planned flights of the Romaeris prototype WSC aircraft will be used to improve the simulation aircraft model. Further refinements of the simulation model will focus on improved approximations for the dynamics of a flexible wing's deforming airfoil. The simulator model will aid in the development of automatic flight controller software specific to weight-shift aircraft.

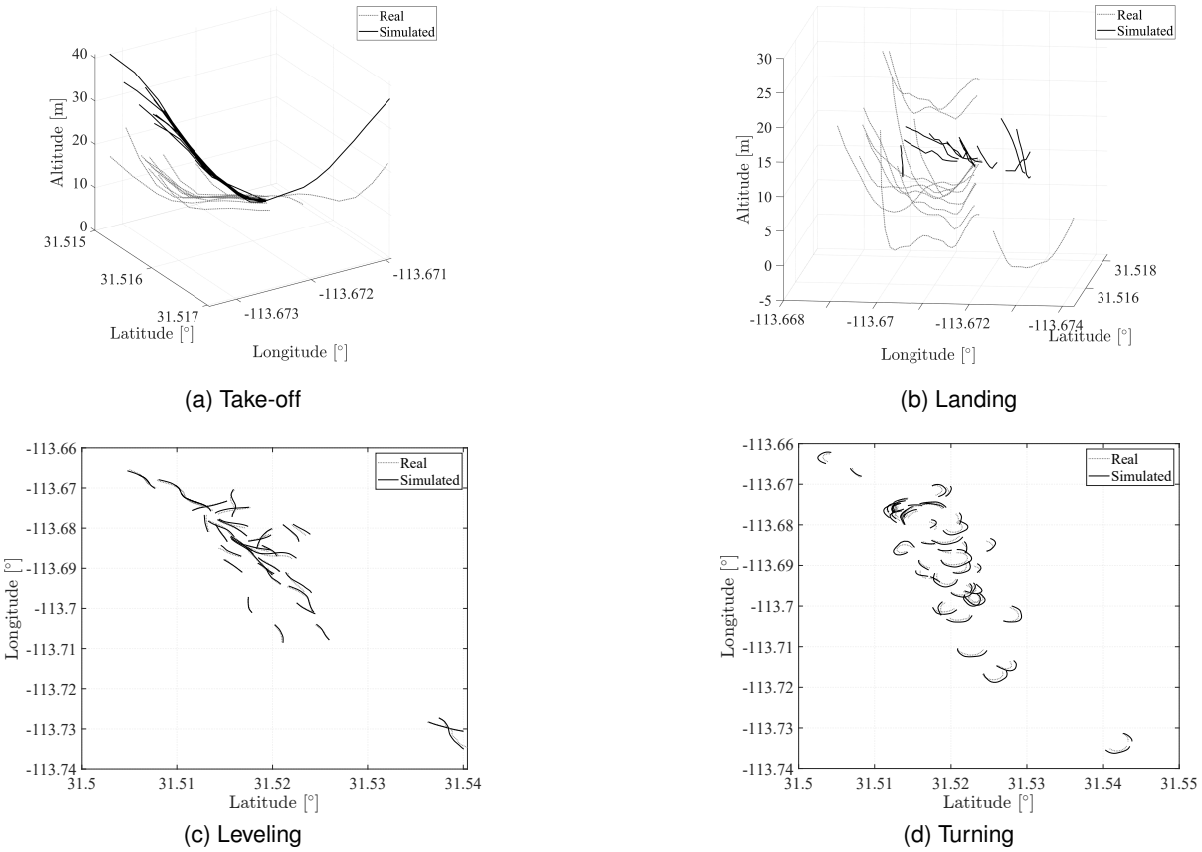


Fig. 6. Flight trajectories according to four flight phase subcategories, comparing the simulated and real flight data.

ACKNOWLEDGMENTS

We are deeply grateful to Romaeris Corporation and Mitacs who provided financial support for this research.

We extend a sincere thanks to Bruno Doerwald of Romaeris Corporation for performing the real-world flight test missions that were central to our simulator benchmarking process, as well as for providing pilot informed reference on the dynamics of weight-shift controlled flight.

REFERENCES

[1] G. B. Gratton, "The weightshift-controlled microlight aeroplane," *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, vol. 215, no. 3, pp. 147–154, 2001.

[2] Federal Aviation Administration, *Weight-Shift Control Aircraft Flying Handbook*. Aviation Supplies & Academics, Inc., 2015.

[3] M. J. Kroes and M. S. Nolan, *Weight-Shift Control Aircraft*, 8th ed., ser. Aircraft Basic Science. New York: McGraw-Hill Education, 2013.

[4] "Romaeris Corporation," https://www.romaeris.com, 2022, accessed: March 6, 2023.

[5] M. Abouheaf, N. Mailhot, and W. Gueaieb, "An Online Reinforcement Learning Wing-Tracking Mechanism for Flexible Wing Aircraft," in *2019 IEEE International Symposium on Robotic and Sensors Environments (ROSE)*, Jun. 2019, pp. 1–7.

[6] M. Abouheaf, N. Q. Mailhot, W. Gueaieb, and D. Spinello, "Guidance Mechanism for Flexible-Wing Aircraft Using Measurement-Interfaced Machine-Learning Platform," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 7, pp. 4637–4648, Jul. 2020.

[7] E. Geller, "Hang glider stability and control," *Technical Soaring*, vol. 27, no. 1-2, pp. 8–17, 2003.

[8] F. M. Rogallo, "Nasa Research on Flexible Wings," *Annals of the New York Academy of Sciences*, vol. 154, no. 2, pp. 953–961, 1968.

[9] P. Dees, "Hang Glider Design and Performance," in *10th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference*. Fort Worth, Texas: American Institute of Aeronautics and Astronautics, September 2010.

[10] E. A. Kilkenny, "An experimental study of the longitudinal aerodynamic and static stability characteristics of hang gliders," phdthesis, Cranfield University, Sep. 1986.

[11] J. T. Sweeting, "An experimental investigation of hang glider stability," Master's thesis, College of Aeronautics, Cranfield University, 1981.

[12] I. Kroo, *Aerodynamics, Aeroelasticity and Stability of Hang Gliders*. Stanford University, 1983.

[13] E. A. Kilkenny, "An evaluation of a mobile aerodynamic test facility for hang glider wings," College of Aeronautics, Cranfield Institute of Technology, Tech. Rep. 8330, 1983.

[14] ——, "Full scale wind tunnel tests on hang glider pilots," Cranfield Institute of Technology, College of Aeronautics, Department of Aerodynamics, Tech. Rep., 1984.

[15] M. V. Cook and E. A. Kilkenny, "An experimental investigation of the aerodynamics of the hang glider," in *Proceedings of an International Conference on Aerodynamics*, 1986.

[16] G. De Matteis, "Response of hang gliders to control," *The Aeronautical Journal*, vol. 94, no. 938, pp. 289–294, 1990.

[17] ——, "Dynamics of hang gliders," *Journal of Guidance Control and Dynamics*, vol. 14, no. 6, pp. 1145–1152, 1991.

[18] W. Khaddage, "A wind-tunnel investigation of an ultra-light wing and ultra-light aircraft," Master's thesis, Carleton University, 2017.

[19] D. M. Blake, "Modelling the aerodynamics, stability and control of the hang glider," Master's thesis, Centre for Aeronautics - Cranfield University, 1991.

[20] M. V. Cook, "The theory of the longitudinal static stability of the hang-glider," *The Aeronautical Journal*, vol. 98, no. 978, pp. 292–304, 1994.

[21] J. Powton, "A theoretical study of the non-linear aerodynamic pitching moment characteristics of the hang glider and its influence on stability and control," mathesis, College of Aeronautics, Cranfield Institute of Technology, 1995.

[22] R. Rollins, "Study of experimental data to assess the longitudinal stability and control of the hang glider," Master's thesis, College of Aeronautics, Cranfield University, 2000.

[23] M. Spottiswoode, "A theoretical study of the lateral-directional dynamics, stability and control of the hang glider," mathesis, College of Aeronautics, Cranfield Institute of Technology, 2001.

[24] M. V. Cook and M. Spottiswoode, "Modelling the flight dynamics of the hang glider," *The Aeronautical Journal*, vol. 109, no. 1102, pp. I–XX, 2005.

[25] M. V. Cook, *Flight Dynamics Principles: A Linear Systems Approach to Aircraft Stability and Control*, 3rd ed., ser. Aerospace Engineering. Butterworth-Heinemann, 2013.

[26] Y. Ochi, "Modeling of the longitudinal dynamics of a hang glider," in *AIAA Modeling and Simulation Technologies Conference*. American Institute of Aeronautics and Astronautics, 2015, pp. 1591–1608.

[27] ——, "Modeling of flight dynamics and pilot's handling of a hang glider," in *AIAA Modeling and Simulation Technologies Conference*. American Institute of Aeronautics and Astronautics, 2017, pp. 1758–1776.

[28] M. Abouheaf, W. Gueaieb, and F. L. Lewis, "Online model-free reinforcement learning for the automatic control of a flexible wing aircraft," *IET Control Theory & Applications*, vol. 14, no. 1, pp. 73–84, 2020.

[29] ——, "Model-Free Gradient-Based Adaptive Learning Controller for an Unmanned Flexible Wing Aircraft," *Robotics*, vol. 7, no. 4, p. 66, Dec. 2018.

[30] M. Abouheaf and W. Gueaieb, "Neurofuzzy Reinforcement Learning Control Schemes for Optimized Dynamical Performance," in *2019 IEEE International Symposium on Robotic and Sensors Environments (ROSE)*, Jun. 2019, pp. 1–7.

[31] ——, "Online model-free controller for flexible wing aircraft: a policy iteration-based reinforcement learning approach," *International Journal of Intelligent Robotics and Applications*, vol. 4, no. 1, pp. 21–43, Mar. 2020. [Online]. Available: https://doi.org/10.1007/s41315-019-00105-3

[32] M. Abouheaf, W. Gueaieb, and D. Spinello, "Online Multi-Objective Model-Independent Adaptive Tracking Mechanism for Dynamical Systems," *Robotics*, vol. 8, no. 4, p. 82, Dec. 2019, number: 4 Publisher: Multidisciplinary Digital Publishing Institute.

[33] M. Abouheaf and W. Gueaieb, "Reinforcement Learning Solution with Costate Approximation for a Flexible Wing Aircraft," in *2018 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA)*, Jun. 2018, pp. 1–6, iSSN: 2377-9322.

[34] M. O. Tokhi and A. K. M. Azad, Eds., *Flexible Robot Manipulators: Modelling, simulation and control*. IET Digital Library, jan 2008. [Online]. Available: https://digital.library.theiet.org/content/books/ce/pbce068e

[35] J. R. Clymer, *Simulation-Based Engineering of Complex Systems*, 2nd ed. Wiley, 2009.

[36] Z. B. Rivera, M. C. De Simone, and D. Guida, "Unmanned Ground Vehicle Modelling in Gazebo/ROS-Based Environments," *Machines*, vol. 7, no. 2, p. 42, Jun. 2019, publisher: Multidisciplinary Digital Publishing Institute.

[37] C. Zhang, Y. Liu, D. Zhao, and Y. Su, "RoadView: A traffic scene simulator for autonomous vehicle simulation testing," in *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, Oct. 2014, pp. 1160–1165, iSSN: 2153-0017.

[38] W. Qian, Z. Xia, J. Xiong, Y. Gan, Y. Guo, S. Weng, H. Deng, Y. Hu, and J. Zhang, "Manipulation task simulation using ROS and Gazebo," in *2014 IEEE International Conference on Robotics and Biomimetics (ROBIO 2014)*, Dec. 2014, pp. 2594–2598.

[39] C. E. Agüero, N. Koenig, I. Chen, H. Boyer, S. Peters, J. Hsu, B. Gerkey, S. Paepcke, J. L. Rivero, J. Manzo, E. Krotkov, and G. Pratt, "Inside the Virtual Robotics Challenge: Simulating Real-Time Robotic Disaster Response," *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 2, pp. 494–506, Apr. 2015.

[40] M. M. Almasri, A. M. Alajlan, and K. M. Elleithy, "Trajectory Planning and Collision Avoidance Algorithm for Mobile Robotics System," *IEEE Sensors Journal*, vol. 16, no. 12, pp. 5021–5028, Jun. 2016.

[41] L. Tai and M. Liu, "A robot exploration strategy based on Q-learning network," in *2016 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, Jun. 2016, pp. 57–62.

[42] W. Li and R. Xiong, "Dynamical Obstacle Avoidance of Task- Constrained Mobile Manipulation Using Model Predictive Control," *IEEE Access*, vol. 7, pp. 88 301–88 311, 2019.

[43] M. M. M. Manhães, S. A. Scherer, M. Voss, L. R. Douat, and T. Rauschenbach, "UUV Simulator: A Gazebo-based package for underwater intervention and multi-robot simulation," in *OCEANS 2016 MTS/IEEE Monterey*, Sep. 2016, pp. 1–8.

[44] N. Imanberdiyev, C. Fu, E. Kayacan, and I.-M. Chen, "Autonomous navigation of UAV by using real-time model-based reinforcement learning," in *2016 14th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, Nov. 2016, pp. 1–6.

[45] S. Suvarna, D. Sengupta, P. Koratikere, and R. S. Pant, "Simulation of Autonomous Airship on ROS-Gazebo Framework," in *2019 Fifth Indian Control Conference (ICC)*, Jan. 2019, pp. 237–241.

[46] A. Ivanovic, M. Polic, J. Tabak, and M. Orsag, "Render-in-the-loop Aerial Robotics Simulator: Case Study on Yield Estimation in Indoor Agriculture," in *2022 International Conference on Unmanned Aircraft Systems (ICUAS)*, Jun. 2022, pp. 787–793, iSSN: 2575-7296.

[47] T. H. Chung, M. R. Clement, M. A. Day, K. D. Jones, D. Davis, and M. Jones, "Live-fly, large-scale field experimentation for large numbers of fixed-wing UAVs," *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1255–1262, May 2016.

[48] D. T. Davis, T. H. Chung, M. R. Clement, and M. A. Day, "Multi-swarm Infrastructure for Swarm Versus Swarm Experimentation," in *Distributed Autonomous Robotic Systems: The 13th International Symposium*, ser. Springer Proceedings in Advanced Robotics, R. Groß, A. Kolling, S. Berman, E. Frazzoli, A. Martinoli, F. Matsuno, and M. Gauci, Eds. Cham: Springer International Publishing, 2018, pp. 649–663.

[49] L. P. M. Nelissen, "Development of Gazebo-based high-fidelity simulation environment for morphing-wing UAVs," Dec. 2021, publisher: University of Twente.

[50] M. Schmittle, A. Lukina, L. Vacek, J. Das, C. P. Buskirk, S. Rees, J. Sztipanovits, R. Grosu, and V. Kumar, "OpenUAV: A UAV Testbed for the CPS and Robotics Community," in *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*, Apr. 2018, pp. 130–139.

[51] R. Mainwaring, "ArduPilot Gazebo Plugin," https://github.com/srmainwaring/ardupilot_gazebo-1, Dec. 2022, accessed: December 1, 2022.

[52] BYU MAGICC Lab, Provo, Utah, "ROS Flight," https://rosflight.org/, accessed: March 15, 2023.

[53] ArduPilot Development Team and Community, "ArduPilot open source autopilot," http://ardupilot.org/, accessed: March 15, 2023.

[54] Open Robotics, "ROS: Robot Operating System," https://www.ros.org, accessed: July 17, 2023.

[55] "Falcon 3 - Wills Wing," https://www.willswing.com/hang-gliders/falcon-3/, accessed: March 15, 2023.

[56] "XFLR5: Analysis tool for airfoils, wings, and planes." http://www.xflr5.tech/xflr5.htm, accessed: July 19, 2023.

[57] "Cosmos aerospace overview," http://meiyagroup.com/cosmos-aerospace.html, accessed: July 19, 2023.

[58] "La mouette microlight wings," https://lamouette.com/en/the-ultralight-wing-ranges/, accessed: July 19, 2023.

[59] Emlid Limited, "Emlid official open source documentation," https://github.com/emlid/emlid-docs, accessed: March 15, 2023.

A.4 Model-Free Force Control of Cable-Driven Parallel Manipulators for Weight-Shift Aircraft Actuation

This is the accepted version of the paper. Published paper DOI: 10.1109/TIM.2023.3346524. ©2023 IEEE. Reprinted, with permission, from Model-free force control of cable-driven parallel manipulators for weight-shift aircraft actuation, IEEE Transactions on Instrumentation and Measurement, 12/2023.

Model-free force control of cable-driven parallel manipulators for weight-shift aircraft actuation

Nathaniel Mailhot, Mohammed Abouheaf, Davide Spinello *Senior Member, IEEE,*

***Abstract*—This paper presents a novel approach to flight control of weight-shift aircraft by employing a cable-driven parallel robot (CDPR) integrated with adaptive force control based on reinforcement learning. Development on weight-shift aircraft control has been sparse. Despite limited but notable efforts, modelling is hindered by parameter uncertainty stemming from highly nonlinear dynamics. The model-free control method introduced in this work operates without knowledge of the complex dynamics problem presented by weight-shift aircraft flight control. An online reinforcement learning technique known as action dependent heuristic dynamic programming (ADHDP) is applied to the problem of coordinating the tension force across parallel cable-driven actuators. Two adaptive learning agents perform demanded weight-shift maneuvers by coordinating torque commands, without an inverse kinematics model. The online reinforcement learning control is implemented on flight controller hardware with limited computational resources and strict timing constraints, performing real-time experiments on a kinematically equivalent surrogate two-body weight-shift mechanism. After online training in the presence of sustained disturbance events, the adaptive learning agents optimally balance against competing trajectory tracking objectives. The CDPR capably reproduces standard S-turn maneuvers, coordinating simultaneous banking and pitching speed actions. The encouraging experimental results inform future integration of the weight-shift CDPRs toward automatic flight control that is unprecedented for this class of aircraft.**

***Index Terms*—adaptive flight control, reinforcement learning, cable-driven parallel robot, hardware implementation.**

I. Introduction

Weight-shift control (WSC) aircraft are characterized by a weight-shifting mass attached to deformable flexible wing used as the primary means of maneuvering, instead of the control surfaces used by standard fixed-wing aircraft [1], [2]. WSC aircraft are appealing as recreational flying sport vehicle because of their simple mechanical structure, ability to operate with little infrastructure on rugged terrain, low cost of fabrication, and high payload-to-mass ratio [3], [4]. WSC aircraft have limited industrial applications due to the requirement for a human pilot. Development of automatic weight-shift flight controls by standard approaches is virtually non-existent due to the significant set of technical challenges, such as the high susceptibility of the flexible wing shape to wind disturbances [5]–[7], control stability in the presence of turbulence [8], [9], and the nonlinear response of weight-shift maneuvers for they depend on the interplay between the vehicle configuration and the flight envelope phase [9]–[12]. This results in high uncertainty for modelling-based control approaches for WSC aircraft, making it a significant challenge with limited, but notable, modelling efforts [13]–[18]. The primary motivation of this work is to employ alternative model-free learning control approaches within the core design and implementation of robotic systems that enable automatic weight-shift flight control to perform trajectory tracking with no input other than sensor feedback. These designs are tested directly on experimental WSC hardware, to prove their suitability for the demands of real-time autonomous flight control for systems of the type schematized in Figure 2.

The low-level weight-shift action is provided by a manipulator composed of cable-winch servo motors attached in a closed kinematic chain, classified as a cable-driven parallel robot (CDPR) [19]–[21]. CDPRs are preferred over open-chain manipulators due to characteristics such-as lower total weight, higher force to weight ratio, larger operating workspace, and high structural stiffness [22]–[24]. In particular, parallel manipulators possess the ability to generate a large force vector on the end-effector for any reachable configuration, so long as the correct set of simultaneous actuator commands can be determined [25], [26]. This represents the critical factor for adopting this manipulator type for WSC problem specifically since the prototype manipulator may approach the same capability of a human pilot using only simple and low cost hardware set-ups, similar to the solutions described for various applications in [27]–[29]. The primary drawback of CDPRs lies in the fact that force on the end effector is applied through applied tension in the set of cables which cannot transmit compressive forces. Hence the control problem becomes complicated, as any candidate controller must generate continuous and simultaneous commands for all actuators in such a way that the sequence of selected levels of tension of cables satisfies the desired end-effector wrench.

The WSC problem for the CDPR is formulated as an approximate dynamic programming (ADP) problem, where the system state is interpreted from sensor feedback to evaluate the performance of an evolving control policy as it performs actions over time within the environment [30], [31]. Model-free learning control methods are suitable for problems with significant parameter uncertainty, which pose an insurmountable challenge for model based approaches even in the presence of accurate models. A temporal difference learning scheme is used to learn optimal control solutions with respect to a user prescribed utility function, using data measured along a sys-

This work was supported by Romaeris Corporation and Mitacs under the Accelerate research fellowship project *Artificial Intelligence (AI) Powered Adaptive Flight Controller for Novel Unmanned Aerial Vehicle (UAV) with Commercial and Humanitarian Applications*.

Nathaniel Mailhot and Davide Spinello are with the Department of Mechanical Engineering, University of Ottawa, Ottawa, Canada. Mohammed Abouheaf is with the School of Engineering, Bowling Green State University, Bowling Green, Ohio, United States of America.

tem's trajectory corresponding to the evolution of the aircraft's state. This process tunes the control gains online and in real-time until the optimal control policy is identified, at which point learning is halted [32]–[35]. To address uncertainty in WSC flight dynamics and provide satisfactory control for the CDPR, servo commands are generated through a model-free adaptive critic scheme by learning agents that are trained to track reference trajectories provided by a high level flight navigation controller.

This work presents two principal contributions. The first is the advancement in the design of intelligent flight controllers for flexible-wing weight-shift aircraft. Initial developments studied parallel linear actuator mechanisms with machine learning applied to an idealized kinematic model of a hang-glider [36], [37]. A model-free guided search learning process based on real-time value iteration was implemented for speed control mechanisms in weight-shift aircraft using an experimental mock-up system with servo-winch actuation in [38]. In contrast, this paper presents for the first time a fully developed model free weight-shift maneuvering scheme based on an actuation system equivalent to a CDPR that mimics the operations of a human weight-shift pilot. To the best of the authors' knowledge, there exists no other control system capable of providing fly-by-wire weight-shift flight control, prior to the solution presented within this work.

The second contribution is the application of online learning control to a CDPR with real-time hardware implementation and testing. Conventional approaches for CDPR control rely upon offline dynamic trajectory planning where an inverse kinematic model is required to solve the coordination problem of the parallel actuators [39]–[42]. In comparison, the approach taken in this work employs an adaptive learning critic scheme to coordinate the parallel manipulators and requires no inverse kinematics model. Optimal CDPR controllers, as seen in [43], [44], while sharing the trait of online optimization, rely on feedback linearization control methods. Approaches such as the model reference adaptive control in [45] or offline deep learning [46] employ neural networks to learn unknown dynamics properties of CDPRs. By comparison, the learning algorithm in this work is completely online and requires no reference model. The selected method is an action dependent heuristic dynamic programming (ADHDP) scheme, similar to Q-learning, designed for online implementation utilizing minimal computational resources while requiring no knowledge of the system, reference, or input dynamics [47]–[49]. Real-time implementation on flight controller hardware attest to the feasibility of the method.

II. Weight-shift flight control mechanism

Weight-shift aircraft control consists of force inputs provided by a pilot or piloting system, to shift the mass of the fuselage relative to the wing, as illustrated in Figures 1 and 2(a). The two bodies, the wing and fuselage, are connected at the mast of the fuselage and the keel tube of the wing by the hang-block joint. The hang-block allows the fuselage to move with two degrees of freedom relative to the wing. The wing system can roll about the longitudinal axis of the keel tube, while the keel bar can pitch about the axis which is perpendicular to both the mast and keel tubes. In crewed flight the pilot situated within the fuselage modifies their orientation with respect to the wing by the reaction from applying force to the control bar attached rigidly to the keel. The centre of gravity (CoG) of the total aircraft is a function of the location of these two primary bodies relative to each-other, and is intentionally modified by the pilot for flight maneuvering. This is illustrated by the example of a banking turn maneuver in Figure 1.

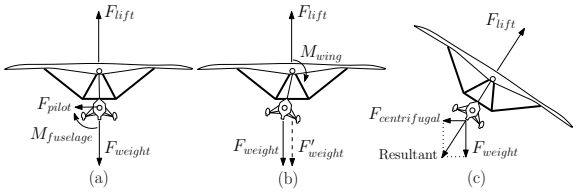


Fig. 1. Free body diagram sequence illustrating the dynamics of the weight-shift aircraft control during a banking roll: (a) Beginning in balance, (b) pilot induced imbalance produces a weight-shifted dynamic equilibrium, (c) resulting in a banking turn when dynamic balance is restored.

A. Schematic of the wing-fuselage piloting system

Weight-shift flight control requires that the pilot enacts the correct force input on the control bar of the wing, given the state of the vehicle relative to its flight envelope. When applying force input on the control bar, the reaction moves the fuselage body, relative to the wing body, modifying the centre of gravity for the total aircraft. The fuselage motion, as well as some small motion of the wing due to the initial reaction to the force input, causes a dynamic imbalance as the CoG of the two bodies become misaligned with respect to gravity. An apparent body moment acts on the wing until this balance is restored, causing a change in the lift vector direction, as illustrated in Figure 1. In the global reference frame defined with gravity pointing downwards, the wing shifts its orientation due to the weight-shift effect. Dynamic weight balancing is the central principle of weight-shift controlled aircraft maneuvering, allowing the aircraft to perform banking turns or wing pitching maneuvers without conventional aerodynamic control surfaces. The pilot or a piloting system modifies the dynamic balance of the aircraft by applying force against the control bar. The reaction of the pilot force moves the fuselage relative to a frame of reference fixed to the wing system, referred to as the wing reference frame. An additional frame of reference is fixed to the fuselage. Notably, the pilot or piloting system must account for the motion of each body, in both reference frames, to correctly perform the intended weight-shift maneuvers, hence has the duty to manage the position, velocity, and acceleration of the fuselage system and the wing system, simultaneously.

A weight-shift aircraft can be schematized by considering the wing and fuselage as two pinned bodies allowed to rotate with respect to each-other in pitch and roll along the hang-block joint. Figure 2(a) presents a schematic for the wing-fuselage system with the CDPR piloting system attached. The

hang-block is equivalent to two attached revolute joints that rotate about two mutually perpendicular axes. The wing body includes the control bar structure, as it is rigidly attached to the wing keel. The fuselage is supported by the wing during flight and is allowed to pitch and roll along the rotation axes of the hang-block. The rotation of the fuselage about the wing keel has a typical allowed roll rotation range of $\theta = \pm 20^\circ$ and a pitch rotation range of $\phi = -10^\circ$ to 15°. The fuselage is constrained in yaw relative to the wing. The wing has no motion constraints and moves freely within six degrees of freedom (DoF) during flight.

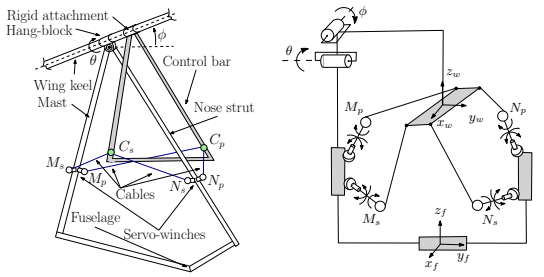


Fig. 2. Schematic (a) of wing-fuselage system with attached CDPR piloting system; (b) the corresponding kinematic diagram.

B. Description of the cable-driven parallel robot

The objective of this work is to design a control mechanism for autonomous flight of flexible wing weight-shift aircraft. The cable-driven parallel robot system replaces the human pilot, allowing for fly-by-wire based weight shift aircraft control. The CDPR consist of four servo-winches that are mounted at four locations along the mast and nose strut on the fuselage frame, i.e. the body that hangs beneath the wing structure. Each servo-winch is attached to the fuselage by a spherical joint, such that the cables maintain alignment with the exit of the winch-spools for any configuration of the robot by reorienting the servo-winches along the tension vector of the cables. Figure 2(b) presents a kinematic diagram of the cable-driven parallel manipulator mechanism. The fuselage is held at a fixed distance from the hang-block, and rotates freely about two axes of rotation, in pitch and roll, relative to the wing frame of reference.

The servo-winches actuate the fuselage by retracting (and releasing) flexible cables that have their ends attached to points on the control bar. Since the control bar is rigidly attached to the wing, any force applied to it is transmitted to the wing. Hence, depending on which servo-winches are actively pulling, the fuselage will move relative to the wing and control bar structure. Therefore the servo-winches control the motion of the fuselage via the parallel tension force vectors.

The weight-shift mechanism allows the fuselage to rotate with two DoF about the wing keel. By using four cables, the robot can be completely restrained for any pose through the selection of a sufficient tension distribution among the four servo-winches. When a cable is under tension, it mechanically transmits the torque produced by its own servo-winch on the control bar in line with the cable alignment. Since the cables can apply only tension forces, intelligent sequencing of servo torque commands is necessary to control the motion of the fuselage.

III. Online model-free solution for the optimal trajectory tracking problem

A model-free trajectory tracking controller for the weight-shift action of the cable-driven parallel robot is designed to overcome high parameter uncertainty and nonlinearity inherent to the system's dynamics. Conventional solution approaches would require knowledge of the system's dynamics to determine the optimal gains offline, backward in time, to apply them in the real time implementation. For linear systems with quadratic performance index, this requires one to explicitly solve Riccati algebraic equation for the gains [50]. Instead, an online ADP approach is employed, which is implemented in forward time and requires only online measurement of the system's trajectory. Given the system's uncertain and nonlinear dynamics, employing the particular ADP technique known as action-dependent heuristic dynamic programming (ADHDP) is appropriate [31], [51]. This method leverages online reinforcement learning to refine estimates for the agent's performance, and approximate the optimal control policy. This solution treats the system dynamics as unknown, deriving all information from measured trajectories and agent actions.

A. Optimal trajectory tracking scheme by online model-free ADHDP

The objective is to define a value function structure that enables the learning agent to observe and be responsive to both the tracking error dynamics and to the reference trajectory dynamics, referred to as trajectory stabilization. The aim is to generate an additional control component equivalent to the feedforward signal inherent to classical trajectory tracking. The modified augmented state vector (MASV) ζ_k is defined as the stack of error dynamics and reference trajectory dynamics,

$$\zeta_k = \begin{bmatrix} e_k \\ (e_k - e_{k-1}) \cdot \Delta_t^{-1} \\ n^{-1}\Delta_t \sum_{i=n}^{\infty}(e_i) \\ r_k \\ (r_k - r_{k-1}) \cdot \Delta_t^{-1} \\ (r_k - 2r_{k-1} + r_{k-2}) \cdot \Delta_t^{-2} \end{bmatrix} \tag{1}$$

where $e_k = x_k - r_k$, is the tracking error measurement, x_k is the state measurement, r_k is the reference trajectory input, and Δ_t is the time step associated with the measurement interval duration. The value function is selected to evaluate system performance across three objectives, namely tracking error minimization, anticipate the reference trajectory dynamics, and minimize the control energy. It is defined as $V_k = \frac{1}{2}\sum_{i=k}^{\infty}\gamma^{i-k}[\zeta_k^T Q \zeta_k + u_k^T R u_k]$, where u_k is the control and Q and R are weight matrices that are designer selected to determine the learning agent's trade-off for each of the different objectives, and the trade off between control energy/effort and tracking accuracy [48]. Moreover, $\gamma \in (0, 1)$ is the discount factor, and V cannot decrease over successive

time steps, i.e. $V_{k+1} - V_k \geq 0$. The summand is the stage cost function, and will be referred to as U_k later on. The discount factor ensures that the learning process does not over value future performance. If the control u_k causes the value function to reach some finite value while stabilizing the system and is finite, it is considered admissible, but not necessarily optimal. The optimal control is obtained when the value function is minimized according to the deterministic Bellman equation $V_k^\star = \min_{u_k}[\zeta_k^T Q \zeta_k + u_k^T R u_k] + \gamma V_{k+1}^\star$. Solving explicitly for the optical control gives $u_{k+1}^\star = -\frac{1}{2}\gamma R^{-1} B^T \nabla V_{k+1}^\star$, where B is the control input map.

B. Actor-critic solution for online value function approximation

The actor-critic solution employed to learn the model-free control online and in real-time for the trajectory stabilization objective is detailed. The Bellman optimality equation acts as a consistency condition, ensuring that the candidate value function follows the optimality principle, converging to the correct desired performance. Two single layer neural networks are trained over time by gradient descent based adaptation laws to approximate the optimal value function (critic unit) and the corresponding controller (actor unit).

The temporal difference (TD) error is zero if Bellman's equation is satisfied, and therefore it is used to quantify the divergence from optimality. In the specific case, the TD error is given by $\epsilon_{\text{TD}} = -V_k + \frac{1}{2}[\zeta_k^T Q \zeta_k + u_k^T R u_k] + \gamma V_{k+1}$. The TD error is applied in a value iteration reinforcement learning scheme to find the optimal control and corresponding value function. Since the TD error depends on the future value function, an approximation scheme consisting on a projection along neural network basis functions. Define the approximate value function as $V_k \approx \hat{V}_k(z_k) = z_k^T \Omega_c z_k / 2$, where $z_k = \begin{bmatrix}\zeta_k & u_k\end{bmatrix}^T$ is the data to be used in the approximation method, $\zeta_k \in \mathbb{R}^n$, $u_k \in \mathbb{R}^m$, and $\Omega_c \in \mathbb{R}^{(n+m)\times(n+m)}$ is a symmetric positive definite weight matrix that must be determined. The initial Ω_c^0 is selected as a random symmetric positive definite with small values such that the initial weight adaptation is more gradual and overall numerical stability is improved during the learning process.

To obtain the optimal control, consider the partition $\Omega_c = \begin{bmatrix}\Omega_{\zeta\zeta} & \Omega_{\zeta u} \\ \Omega_{u\zeta} & \Omega_{uu}\end{bmatrix}$, where $\Omega_{\zeta\zeta} \in \mathbb{R}^{n\times n}$, $\Omega_{\zeta u} = \Omega_{u\zeta}^T$, $\Omega_{\zeta u} \in \mathbb{R}^{n\times m}$, and $\Omega_{uu} \in \mathbb{R}^{m\times m}$. Should the optimal $\Omega_c^\star$ be known, the corresponding optimizing control is given by $u_k^\star = \Omega_\pi^\star \zeta_k = -\Omega_{uu}^{\star -1}\Omega_{u\zeta}^\star \zeta_k$, where $\Omega_\pi^\star$ is the policy determined by the optimality condition. During the learning process, a probing white noise is added to the control input signal to ensure that the persistent excitation condition is satisfied [48], [52]. When the learning algorithm converges to the optimal policy, the probing noise is turned off and the learning is halted.

An adaptive critic scheme is employed to identify the critic weight matrix Ω_c. The critic neural network weights are evaluated by calculating the temporal difference of the value function estimate

$$\epsilon_{\text{TD}} = -\frac{1}{2} z_k^T \Omega_c z_k + U_k + \gamma \frac{1}{2} z_{k+1}^T \Omega_c z_{k+1} \tag{2}$$

With each iteration step, the critic weights are updated to minimize the square of the TD error, resulting in the adaptive update

$$\Omega_c^{j+1} = \Omega_c^j - \alpha_c \epsilon_{TD} z_k z_k^T \tag{3}$$

where α_c is the critic learning rate. The index j tracks the version of Ω_c. The critic weights Ω_c could be used to calculate the optimal policy Ω_π directly as $\Omega_\pi^\star \zeta_k = -\Omega_{uu}^{\star -1}\Omega_{u\zeta}^\star \zeta_k$. However, this approach is not advised as the critic weights have the potential to rapidly change during the online adaptation. Since the critic is merely an approximation of the value function, this can very well result in consistently poor control policy. A second adaptive actor neural network Ω_a is introduced with actor weights updated according to the adaptive update law

$$\Omega_a^{j+1} = \Omega_a^j - \alpha_a(\Omega_a^j \zeta_k - (-\Omega_{uu}^{j+1})^{-1}\Omega_{u\zeta}^{j+1}\zeta_k)\zeta_k^T \tag{4}$$

where α_a is the actor learning rate. The corresponding optimal control is now given by

$$u_k = \Omega_a^\star \zeta_k \tag{5}$$

and $\Omega_a^\star$ is the feedback policy found after convergence of the actor neural network adaptation. The actor-critic adaptation laws provide a means to approximate the trajectory stabilized tracking value function online and in real time, and in turn provides the optimal control solution. Algorithm 1 summarizes the online value iteration actor-critic approach that is used. Figure 3 provides the complementary control system diagram.

Algorithm 1 Online value iteration with actor-critic structure

1: Initialize state measurements x_{0-2}, reference trajectory states r_{0-2} critic Ω_c^0 as random symmetric P.D. with small values , and actor Ω_a^0.
2: Calculate ζ_0 using (1)
3: **begin learning process**
4: **repeat**
5: Update ζ_k using (1)
6: Calculate u_k using (5), apply to system
7: Allow one time step to pass, $k \leftarrow k+1$
8: Update ζ_{k+1} using (1)
9: Calculate TD error ϵ_{TD} using (2)
10: Apply critic update law (3) to find Ω_c^{j+1}
11: Apply actor update law (4) to find Ω_a^{j+1}
12: **until** actor gains have converged and satisfactory trajectory tracking performance is achieved
13: **halt learning process**
14: Calculate further control input with the optimal control $u_k^\star = \Omega_a^\star \zeta_k$

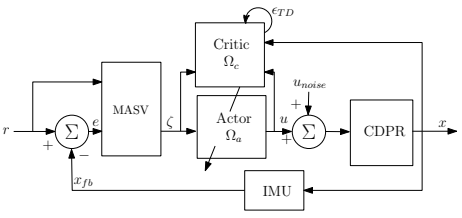


Fig. 3. Trajectory stabilizing actor-critic model-free controller.

C. Implementation of weight-shift control on the CDPR with multi-agent learning

As discussed in Section II, the goal of WSC is to control the attitude of the wing ϕ_w and θ_w, denoting the wing pitch and roll respectively. During in-flight operations the required

performance criteria for the pitch control action may be different than that of the roll control action. The CDPR introduced in Section II-B applies force input to the fuselage by sending simultaneous torque commands across the set of servo-winch actuators to manipulate the wing attitude. Separate commands are necessary to provide the required flexibility for pitch and roll flight controls, hence two separate learning agents are implemented using Algorithm 1. Each agent is provided the commanded reference trajectory and IMU measurement for a single element of the wing attitude, i.e. $r_k^{(1)} = \phi_w^{cmd}$, $r_k^{(2)} = \theta_w^{cmd}$ and $x_k^{(1)} = \phi_w$, $x_k^{(2)} = \theta_w$, and produces its own control input, denoted as $u_k^{(1)}$ and $u_k^{(2)}$. The control inputs are then mixed according to a simple heuristic that depends on the arrangement of the servo-winches. The multi-agent system diagram is presented in Figure 4.

TABLE I
TORQUE DIVISION LOGIC, MARKS INDICATE THE SERVOS ACTIVATED FOR THE CORRESPONDING ATTITUDE CORRECTION.

	$u^{(1)}:+\phi_w$	$u^{(1)}:-\phi_w$	$u^{(2)}:+\theta_w$	$u^{(2)}:-\theta_w$
M_s	✓	...	✓	...
M_p	✓	...	...	✓
N_s	...	✓	✓	...
N_p	...	✓	...	✓

The arrangement of the servo-winch mounts is used to select a heuristic for the relationship between the active servo-winch's and the the corresponding state change for ϕ_w, θ_w. As illustrated in Figure 2(a), the actuators M_s, M_p, N_s, N_p are positioned relative to the end effector attachment points C_s, C_p. The heuristic logic is summarized by Table I. For example, a demanded pitch increase $+\phi_w$ with no roll action implies that the first agent acts, $u_k^{(1)}$ is increased and the demanded torque magnitude is partitioned between M_s and M_p, while the second agent does not act, i.e. $u_k^{(2)}$ does not change. For a second example where a simultaneous demanded pitch increase $+\phi_w$ and a demanded roll decrease $-\theta_w$ occurs, the first agent acts by increasing $u_k^{(1)}$ and the torque is divided between M_s and M_p, while the second agent acts by decreasing $u_k^{(2)}$ and the torque magnitude is divided between M_p and N_p. In the given second example the net demanded torque for M_p is applied as it is being shared by both agents simultaneously. The experimental results within the following section will illustrate the interplay inherent to the parallel actuator system.

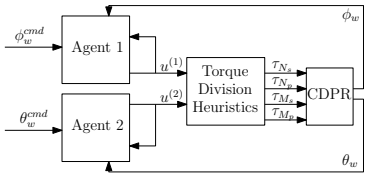


Fig. 4. Multi-agent learning controller for the cable-driven parallel robot. The agents share the set of servomotors to actuate the weight-shift mechanism.

IV. EXPERIMENTAL RESULTS AND DISCUSSION

The proposed model-free learning control is validated experimentally using real flight controller hardware with limited computational resources and strict control update constraints. A prototype CDPR mounted to a surrogate weight-shift mechanism tracks reference wing trajectories ϕ_w, θ_w by shifting the fuselage weight. The selected maneuvers represent repeated S-turns of a real weight-shift aircraft, a standard training maneuver as informed by the partner's test pilot [1].

A. Real-time learning control with weight-shift mechanism

The experimental platform shown in Figure 5 reproduces the kinematics to the weight-shift controlled aircraft illustrated in Figure 2. The wing analogue is composed of a longitudinal member representing the wing's keel, with a triangle frame rigidly attached to the keel that represents the pilot control bar. The weight-shift fuselage is suspended from the wing keel by a two DoF mechanical joint that is analogous to the hang-block. The wing keel rotates relative to hang point in a range of $\pm 8^\circ$ in pitch and $\pm 10^\circ$ in roll, corresponding to the fuselage displacement relative to the keel of $\pm 20^\circ$ in roll and -10° to 15°. The control of the platform is actuated by the CDPR. Actuation is provided by four Dynamixel XM-430 servomotors [53] with integrated cable winches. These manipulators are mounted to different positions along the fuselage as illustrated in Figure 2(a), and have the other end of their cables attached at positions on the control bar. The servomotors receive current input commands at a rate of $50\,\text{Hz}$, i.e. control update interval $\Delta_t = 20\,\text{ms}$. The adaptation for both critic and actor neural networks is computed within this control update interval directly on the flight controller hardware according to (3) and (4).

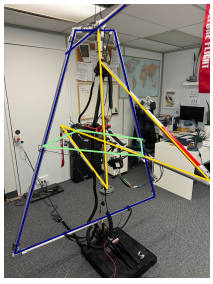


Fig. 5. Experimental weight-shift control mock-up used to emulate the weight-shift effect of the wing controlled by the CDPR. The wing keel and control bar (blue) are suspended at the hang-block, and are rotated by weight-shift of the fuselage (yellow) by forces on the parallel cables (green).

Each hardware device and supporting software is either open-source or has publicly available documentation. An Emlid Navio2 [54] hat board attached to a Raspberry Pi 3 model B, this unit possessing a 1.2GHz 64-bit quad-core ARMv8 CPU, 1GB of RAM and a Cortex-M3 co-processor. [55] is selected as the controller's computational unit due its prevalence as one hardware of choice for open-source flight controller platforms like ArduPilot [56]–[58]. Navio2

provides redundant orientation estimates through the combination of sensor readings from either the main on-board InvenSense MPU-9250 9 DoF Inertial Measurement Unit (IMU) [59] or the secondary Microelectronics LSM9DS1 9 DoF IMU [60]. Each IMU has a three-axis magnetometer, three-axis gyroscope, and three-axis accelerometer. The IMU signals are filtered by a programmable digital low pass filter, which is routed to the Raspberry Pi 3. The total root mean square noise of the MPU-9250 used for inertial measurement feedback is provided to be $0.1\,^{\circ}/\mathrm{s}$. The measurements data are filtered by a low-pass filter with a cut-off frequency of $250\,\mathrm{Hz}$. The resulting angular position measurements of the wing orientation are sampled at $20\,\mathrm{Hz}$ by the online model-free control.

The learning agents determine the correct control by adjusting theirs actions according to measurements of the past trajectory tracking performance. The trained controller then selects the set of demanded torque commands for all actuators simultaneously. A controller agent is trained for each component of the trajectory. In each experiment the training is halted after $50\,\mathrm{s}$ and the probing noise is removed. The performance of the converged control policy is observed for an additional $50\,\mathrm{s}$. The set of parameters in Table II are held constant across all experiments. U and N denote the uniform and normal distributions.

TABLE II
LEARNING CONTROLLER EXPERIMETNAL PARAMETERS.

Parameter	Value	Description
Q	$10 \cdot I_6$	state weighting
R	1	control weighting
$\Omega_a^{(i)0}$	$[20, 20, 20, 10, 10, 10]$	initial actor
$\Omega_c^{(i)0}$	$U(0.001)_{7\times 7},\ \Omega_c^0 > 0$	initial critic
u_{noise}	$N(0, 0.5)$	probing noise
Δ_t	20 ms	update interval
α_c	0.01	critic learning rate
α_a	0.001	actor learning rate
γ	0.999	discount factor

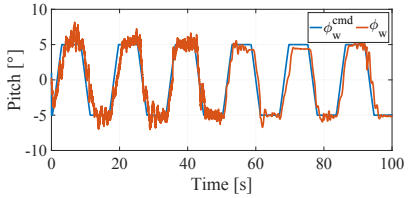
B. Tracking of weight-shift S-turn commands

For the first sample result, ϕ_w^{cmd} and θ_w^{cmd} evolve according to the saturating sinusoidal signals that define a repeated weight-shift S-turn, given by

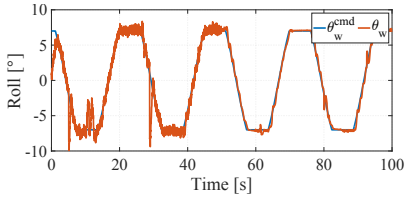
$$\phi_w^{cmd} = \begin{cases} 10\sin(0.06\pi k), & \text{if} |r_k| \leq 5^{\circ} \\ 5^{\circ}, & \text{otherwise} \end{cases} \tag{6a}$$

$$\theta_w^{cmd} = \begin{cases} 10\sin(0.04\pi k), & \text{if} |r_k| \leq 7^{\circ} \\ 7^{\circ}, & \text{otherwise} \end{cases} \tag{6b}$$

with the goal of demonstrating higher amplitude roll bank commands coordinated with pitch speed control commands. The compound pitch and roll tracking performance of the wing is shown in Figure 6. After training is completed, the controller anticipates and tracks the demanded reference trajectory for both axes by application of fuselage weight-shift. When both reference trajectory components change at the same time a transient error is observed as the system adjust each axis in response. After training ends, the actor neural network gains for each agent are $\Omega_a^{(1)} = \left[3.43, 0.26, 9.80, 15.36, 9.03, 12.26\right]$ and $\Omega_a^{(2)} = \left[6.59, 0.14, 11.49, 15.36, 11.04, 13.20\right]$. Figure 7 shows the normalized servo torque commands that were generated by the controller during the experiment. It is observed that as the training progresses, the demanded torque variation and amplitudes decrease as the controller learns the optimal policy. Once training ends, the controller generates the correct combination of torque commands for all four servo-motors simultaneously, presented in Figure 7b.

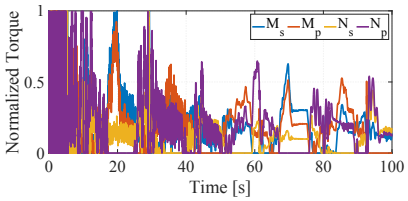


(a) ϕ_w

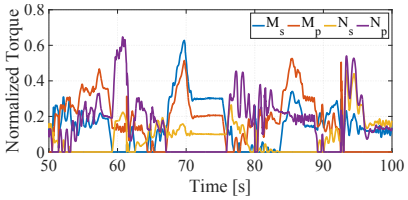


(b) θ_w

Fig. 6. Trajectory tracking of wing's pitch and roll for coordinated S-turns.



(a) 0 - 100 s

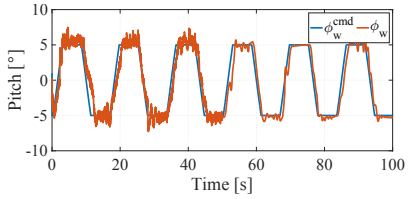


(b) 50 - 100 s, (post-training)

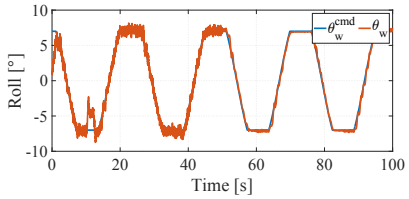
Fig. 7. Normalized servo torque commands.

To test the adaptability of the learning process, the second experiment applies a sustained disturbance event during training. Disturbances are implemented as random white noise signal sampled from the normal distribution $N(2, 0.5)$ added to the IMU feedback at each time step between $10\,\mathrm{s}$ to $15\,\mathrm{s}$, corresponding to 10% of the training period. In the experiment, the disturbances induce a significant tracking error with a

magnitude as high as $\pm 5^\circ$. The learning control would rarely converge beyond this magnitude of sustained disturbance. The reference trajectory is kept the same as the prior sample, i.e. evolving according to (6). Figure 8 shows the tracking performance when the sustained disturbance is present. The controller tracking recovers quickly after the disturbance ends. The trajectory tracking performance achieved after the training ends is unaffected. The disturbance rejection property of the learning algorithm is being tested up to the level of disturbance stated below; further investigation is required to establish if it holds under more extreme events of longer duration or high amplitude.



(a) ϕ_w



(b) θ_w

Fig. 8. Trajectory tracking with sustained disturbance during learning, beginning at 10 s.

C. Summary of experimental results

The CDPR system is found to be a viable candidate automatic weight-shift control actuation scheme as it demonstrates the correct kinematic and force interface to match a human pilot performing repeating S-turn maneuvers with coordinated pitch speed input control. The ADHDP technique proves to be viable as a means to implement an online reinforcement learning approach with computationally constrained flight controller hardware. As no model for the system dynamics is known in advance, the adaptive learning agent determines control policy according to the heuristic it is given as presented by Table I. In each experiment, the learning agent identifies an optimal policy in real-time to balance trajectory tracking error minimization, anticipate reference trajectory changes, and minimize the energy used by the servo-torques. Further, it is found that the learning agents can determine an optimal policy even when a sustained disturbance event is present during the learning phase, resulting in a similar set of control gains for both experiments.

V. Conclusion

A solution to automatic weight-shift aircraft control is realized through the application of a cable-driven parallel robot that is controlled by a model-free controller based on reinforcement learning. The ADHDP technique allows for an online learning controller to be directly implemented on real flight controller hardware that is computationally limited and requires strict control updates timing. Real-time experiments validate the learning controllers performance by demonstrating its ability to perform weight-shift maneuvers trajectory tracking through the coordination of force applied by the parallel actuators. To the best of the authors' knowledge, this represents the first time a viable automatic weight-shift control mechanism for this class of aircraft has been reported on. The piloting mechanism has full authority over the entire range of weight-shift control, matching the kinematic capability of a human pilot.

Ongoing work includes the replacement of the torque division heuristic to distribute the action of the winches with an optimization scheme based on minimal energy consumption. Additionally, work is underway to test the presented solution with simulated flight with a novel simulation model that approximates the weight-shift aircraft flight dynamics and can be tested with standard open-source robot simulation tools.

Acknowledgments

We are grateful to Romaeris Corporation and Mitacs for providing financial support for this research. We extend a special thanks to Riyanson Alfred from Romaeris for providing technical assistance with the embedded systems used for the experiments.

References

[1] Federal Aviation Administration, *Weight-Shift Control Aircraft Flying Handbook*. Aviation Supplies & Academics, Inc., 2015.
[2] S. Salahudden, V. S. Dwivedi, and A. K. Ghosh, "Roll angle optimization in coordinated level turn flight and its analytical validation for UAV," in *IEEE Aerospace Conference*, Mar. 2019, pp. 1–6.
[3] G. B. Gratton, "The weightshift-controlled microlight aeroplane," *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, vol. 215, no. 3, pp. 147–154, 2001.
[4] P. Dees, "Hang glider design and performance," in *AIAA Aviation Technology, Integration, and Operations Conference*. Fort Worth, Texas: American Institute of Aeronautics and Astronautics, September 2010.
[5] Y. Ochi, "Modeling of the longitudinal dynamics of a hang glider," in *AIAA Modeling and Simulation Technologies Conference*, 2015, pp. 1591–1608.
[6] M. Spottiswoode, "A theoretical study of the lateral-directional dynamics, stability and control of the hang glider," M.Sc. thesis, College of Aeronautics, Cranfield Institute of Technology, 2001.
[7] Y. Ochi, "Modeling of flight dynamics and pilot's handling of a hang glider," in *AIAA Modeling and Simulation Technologies Conference*, 2017, pp. 1758–1776.
[8] J. T. Sweeting, "An experimental investigation of hang glider stability," M.Sc. thesis, College of Aeronautics, Cranfield Institute of Technology, 1981.
[9] M. V. Cook, "The theory of the longitudinal static stability of the hang-glider," *The Aeronautical Journal*, vol. 98, no. 978, pp. 292–304, 1994.
[10] E. A. Kilkenny, "Full scale wind tunnel tests on hang glider pilots," Cranfield Institute of Technology, College of Aeronautics, Department of Aerodynamics, technical report, 1984.
[11] E. A. Kilkenny, "An experimental study of the longitudinal aerodynamic and static stability characteristics of hang gliders," Ph.D. dissertation, Cranfield Institute of Technology, Sep. 1986.
[12] D. M. Blake, "Modelling the aerodynamics, stability and control of the hang glider," M.Sc. thesis, Centre for Aeronautics - Cranfield Institute of Technology, 1991.
[13] I. Kroo, "Aerodynamics, aeroelasticity and stability of hang gliders," Stanford University and NASA Ames Research Center, Moffett Field, California, technical report, 1981.

[14] G. De Matteis, "Response of hang gliders to control," *The Aeronautical Journal*, vol. 94, no. 938, pp. 289–294, 1990.

[15] G. De Matteis, "Dynamics of hang gliders," *Journal of Guidance Control and Dynamics*, vol. 14, no. 6, pp. 1145–1152, 1991.

[16] M. V. Cook and M. Spottiswoode, "Modelling the flight dynamics of the hang glider," *The Aeronautical Journal*, vol. 109, no. 1102, pp. 1–20, 2005.

[17] W. Khaddage, "A wind-tunnel investigation of an ultra-light wing and ultra-light aircraft," M.A.Sc. thesis, Carleton University, 2017.

[18] E. Lanteigne, J. McLeod, M. Vadsola, and S. Liu, "On the design of structural wing members for an unmanned weight-shift aircraft," *Journal of Unmanned Vehicle Systems*, vol. 8, no. 3, pp. 161–171, 2020.

[19] T. Paty, N. Binaud, S. Caro, and S. Segonds, "Cable-driven parallel robot modelling considering pulley kinematics and cable elasticity," *Mechanism and machine theory*, vol. 159, pp. 104 263–, 2021.

[20] J. Peng, W. Xu, F. Wang, Y. Han, and B. Liang, "A hybrid hand-eye calibration method for multilink cable-driven hyper-redundant manipulators," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–13, 2021.

[21] M. H. Vu and U. J. Na, "A new 6-DOF haptic device for teleoperation of 6-DOF serial robots," *IEEE Transactions on Instrumentation and Measurement*, vol. 60, no. 11, pp. 3510–3523, Nov. 2011.

[22] M. Zarebidoki, J. S. Dhupia, and W. Xu, "A review of cable-driven parallel robots: Typical configurations, analysis techniques, and control methods," *IEEE Robotics & Automation Magazine*, vol. 29, no. 3, pp. 89–106, 2022.

[23] I. Ebert-Uphoff and P. A. Voglewede, "On the connections between cable-driven robots, parallel manipulators and grasping," in *Proceedings IEEE International Conference on Robotics and Automation*, vol. 5, 2004, pp. 4521–4526 Vol.5.

[24] H. A. F. Almurib, H. F. Al-Qrimli, and N. Kumar, "A review of application industrial robotic design," in *2011 Ninth International Conference on ICT and Knowledge Engineering*, 2012, pp. 105–112.

[25] Y. Yiu, H. Cheng, Z. Xiong, G. L. Liu, and Z. Li, "On the dynamics of parallel manipulators," in *Proceedings IEEE International Conference on Robotics and Automation*, vol. 4, 2001, pp. 3766–3771 vol.4.

[26] H. Khakpour, L. Birglen, and S.-A. Tahan, "Synthesis of differentially driven planar cable parallel manipulators," *IEEE Transactions on Robotics*, vol. 30, no. 3, pp. 619–630, 2014.

[27] F. Pierrot, V. Nabat, O. Company, S. Krut, and P. Poignet, "Optimal design of a 4-dof parallel manipulator: From academia to industry," *IEEE Transactions on Robotics*, vol. 25, no. 2, pp. 213–224, 2009.

[28] N. Mostashiri, J. S. Dhupia, A. W. Verl, and W. Xu, "A review of research aspects of redundantly actuated parallel robotsw for enabling further applications," *IEEE/ASME Transactions on Mechatronics*, vol. 23, no. 3, pp. 1259–1269, 2018.

[29] S. Qian, B. Zi, W.-W. Shang, and Q.-S. Xu, "A review on cable-driven parallel robots," *Chinese journal of mechanical engineering*, vol. 31, no. 1, pp. 29–39, 2018.

[30] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. Massachusetts: MIT Press, 1998.

[31] P. J. Werbos, "Neurocontrol and supervised learning: An overview and evaluation," in *Handbook of Intelligent Control: Neural, Fuzzy, and Adaptive Approaches*, D. A. White and D. A. Sofge, Eds. Van Nostrand Reinhold, Jun. 1992, pp. 65–89.

[32] M. I. Abouheaf, F. L. Lewis, K. Vamvoudakis, S. Haesaert, and R. Babuska, "Multi-agent discrete-time graphical games and reinforcement learning solutions," *Automatica*, vol. 50, no. 12, pp. 3038–3053, 2014.

[33] M. Abouheaf and W. Gueaieb, "Reinforcement learning solution with costate approximation for a flexible wing aircraft," in *IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications*, Jun. 2018, pp. 1–6.

[34] M. Abouheaf and W. Gueaieb, "Online model-free controller for flexible wing aircraft: a policy iteration-based reinforcement learning approach," *International Journal of Intelligent Robotics and Applications*, vol. 4, no. 1, pp. 21–43, Mar. 2020.

[35] M. Abouheaf, W. Gueaieb, and F. L. Lewis, "Online model-free reinforcement learning for the automatic control of a flexible wing aircraft," *IET Control Theory & Applications*, vol. 14, no. 1, pp. 73–84, 2020.

[36] M. I. Abouheaf, W. Gueaieb, and F. L. Lewis, "Model-free gradient-based adaptive learning controller for an unmanned flexible wing aircraft," *Robotics*, vol. 7, no. 4, p. 66, 2018.

[37] M. Abouheaf, N. Mailhot, and W. Gueaieb, "An online reinforcement learning wing-tracking mechanism for flexible wing aircraft," in *IEEE International Symposium on Robotic and Sensors Environments*, June 2019, pp. 1–7.

[38] M. Abouheaf, N. Q. Mailhot, W. Gueaieb, and D. Spinello, "Guidance mechanism for flexible-wing aircraft using measurement-interfaced machine-learning platform," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 7, pp. 4637–4648, 2020.

[39] X. Jiang and C. Gosselin, "Dynamic point-to-point trajectory planning of a three-dof cable-suspended parallel robot," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1550–1557, Dec. 2016.

[40] N. Zhang and W. Shang, "Dynamic trajectory planning of a 3-DOF under-constrained cable-driven parallel robot," *Mechanism and Machine Theory*, vol. 98, pp. 21–35, Apr. 2016.

[41] B. Zhang, W. Shang, S. Cong, and Z. Li, "Coordinated dynamic control in the task space for redundantly actuated cable-driven parallel robots," *IEEE/ASME Transactions on Mechatronics*, vol. 26, no. 5, pp. 2396–2407, Oct. 2021.

[42] D. Lin and G. Mottola, "Dynamic launch trajectory planning of a cable-suspended translational parallel robot using point-to-point motions," *Machines*, vol. 11, no. 2, p. 224, Feb. 2023.

[43] H. R. Fahham and M. Farid, "Minimum-time trajectory planning of spatial cable-suspended robots along a specified path considering both tension and velocity constraints," *Engineering Optimization*, vol. 42, no. 4, pp. 387–402, Apr. 2010.

[44] M. H. Korayem, H. Tourajizadeh, A. Zehfroosh, and A. H. Korayem, "Optimal path planning of a cable-suspended robot with moving boundary using optimal feedback linearization approach," *Nonlinear Dynamics*, vol. 78, no. 2, pp. 1515–1543, Oct. 2014.

[45] H. Jabbari Asl and F. Janabi-Sharifi, "Adaptive neural network control of cable-driven parallel robots with input saturation," *Engineering Applications of Artificial Intelligence*, vol. 65, pp. 252–260, Oct. 2017.

[46] H. Xiong, L. Zhang, and X. Diao, "A learning-based control framework for cable-driven parallel robots with unknown jacobians," *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 234, no. 9, pp. 1024–1036, 2020.

[47] A. Al-Tamimi, M. Abu-Khalaf, and F. L. Lewis, "Adaptive critic designs for discrete-time zero-sum games with application to H_∞ control," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 37, no. 1, pp. 240–247, Feb. 2007.

[48] F. L. Lewis, D. Vrabie, and V. Syrmos, *Optimal Control*, 3rd ed. New York, USA: Wiley, 2012.

[49] B. Kiumarsi, F. L. Lewis, H. Modares, A. Karimpour, and M.-B. Naghibi-Sistani, "Reinforcement Q-learning for optimal tracking control of linear discrete-time systems with unknown dynamics," *Automatica*, vol. 50, no. 4, pp. 1167–1175, Apr. 2014.

[50] R. E. Bellman, *Dynamic Programming*. Princeton University Press, 1957.

[51] T. Landelius and H. Knutsson, "Greedy adaptive critics for LQR problems: Convergence proofs," *Neural Computation*, Jan. 1996.

[52] D. Vrabie, O. Pastravanu, M. Abu-Khalaf, and F. L. Lewis, "Adaptive optimal control for continuous-time linear systems based on policy iteration," *Automatica*, vol. 45, no. 2, pp. 477–484, Feb. 2009.

[53] Robotis Co., "Dynamixel SDK," accessed: March 15, 2023. [Online]. Available: http://emanual.robotis.com/docs/en/software/dynamixel

[54] Emlid Limited, "Emlid official open source documentation," accessed: March 15, 2023. [Online]. Available: https://github.com/emlid/emlid-docs

[55] Raspberry Pi Foundation, "Raspberry Pi official open source documentation," https://github.com/raspberrypi/documentation, accessed: March 15, 2023.

[56] ArduPilot Development Team and Community, "ArduPilot open source autopilot," accessed: March 15, 2023. [Online]. Available: http://ardupilot.org/

[57] S. Tabor, I. Guilliard, and A. Kolobov, "Ardusoar: An open-source thermalling controller for resource-constrained autopilots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2018, pp. 6255–6262.

[58] H. A. de Oliveira and P. F. Ferreira Rosa, "Genetic neuro-fuzzy approach for unmanned fixed wing attitude control," in *International Conference on Military Technologies*, May 2017, pp. 485–492.

[59] TDK Corp. InvenSense, "TDK InvenSense MPU-9250 sensor documentation," accessed: December 1, 2022. [Online]. Available: https//www.invensense.com/products/motion-tracking-9-axis/mpu-9250/

[60] STMicroelectronics N.V., "STM LSM9DS1 sensor documentation," accessed: January 15th. [Online]. Available: https://www.st.com/en/mems-and-sensors/lsm9ds1.html

A.5 Automatic Weight-Shift Aircraft Control with Experimental Fly-By-Wire Flights

Automatic Weight-Shift Aircraft Control with Experimental Fly-By-Wire Flights

Nathaniel Mailhot*, Teresa de Jesus Krings†, Davide Spinello*
*Department of Mechanical Engineering, University of Ottawa, Ottawa, ON, Canada
†Department of Mechanical and Aerospace Engineering, Carleton University, Ottawa, ON, Canada
Email: nmailhot@uottawa.ca, teresadejesuskrings@cmail.carleton.ca, dspinell@uottawa.ca

***Abstract*—Weight-shift control (WSC) ultralights are a class of aircraft that maneuver by manipulating a mass attached to deformable flexible wings, which are favoured by many recreational pilots due to their simple mechanical structure, superior low-speed maneuverability, positive aerodynamic stability, and capability to operate on rugged terrains with minimal infrastructure requirements for take-off and landing. However, the dependency on specialized human-powered physical control restricts their effective payload and mission scope. This work presents experimental validation of automatic electronic flight control for ultralight weight-shift aircraft by application of a novel actuation design that consists of a cable-driven parallel robot (CDPR). After validation of the CDPR's capabilities within a laboratory settings to produce trajectory tracking for different weight-shifting scenarios, the prototype actuation system is integrated with a human-piloted weight-shift aircraft as an electronic fly-by-wire control for attitude adjustment. The test pilot uses a joystick control scheme to perform weight-shift flight maneuvers. The results illustrate the CDPR's ability to supplant conventional human-induced weight-shift operations and lays the groundwork for autonomous flight control for weight-shift aircraft.**

***Index Terms*—automatic flight control, cable-driven parallel robot, ultralight flexible-wing aircraft, hardware implementation, electronic fly-by-wire systems, human-machine interaction**

I. Introduction

Weight-shift control (WSC) ultralight aircrafts maneuver by manipulating a mass attached to flexible wings. They are adopted for their simple structure and low-speed maneuverability, while operating in rugged terrains with minimal infrastructure [1]–[3]. However, the need for physically laborious, very specialized, and manual human-powered control restricts their effective payload and mission scope. As a result, WSC aircraft are primarily used for recreation by enthusiast pilots, with limited industrial applications. This work introduces a novel fly-by-wire (FBW) electronic weight-shift control piloting system that enables intelligent robotic control, eliminating the physical requirement on the human pilot, thereby improving safety and opening the door to further flight automation.

Flight control development for weight-shift aircraft, still in its infancy, suffers from limited research on dynamic modelling. Creating accurate dynamic models is complex due to the nonlinear dynamics caused by airfoil deformation in the flexible wing and the inherent two-body weight-shift relationship in their flight control [4]. As such, the existing dynamics models possess a high degree of parameter uncertainty. The design history of flexible wing WSC aircraft that began with Rogallo's innovations aimed at space vehicle recovery, later transitioned to recreational flying where design evolution proceeded without rigorous scientific methodology, resulting in elevated accident rates [5]–[7]. Early aerodynamic models were formulated using data from ground vehicle-towed flying rigs, leading to insights on aeroelastic and aerodynamic stability properties [8], [9]. Kilkenny and Kroo separately worked to develop hang-glider wing aerodynamic models through wind-tunnel and mobile flying rig experiments, providing a foundation for understanding longitudinal dynamics [7], [9]–[11]. Concurrently, preliminary dynamic models by de Matteis considered rigid wings and ignored the velocity-dependent deformation, recently validated through scaled wind-tunnel tests by Khaddage [12]–[14].

Cook et al.'s mathematical model describes hang-glider kinetics through nonlinear dynamics of inertial, gravitational, and aerodynamic forces, including trim equilibrium [15]. These equations capture small disturbances in motion variables as stability and control derivatives, using classical flight control techniques [16]. The intricate part of this model involves the aerodynamic modelling of the deforming flexible wing by detailing how deformations in luff, camber, and twist affect the aircraft's aerodynamic properties and overall stability. However, the model is limited by its quasi-static focus around equilibrium trim conditions, neglecting non-steady aerodynamic effects and the impact of the pilot's weight shifting. Ochi extended this work to include equations of motion for the two-body system of hang-glider wing with the pilot that is constrained by a holonomic strap, and explored how pilot control inputs can stabilize the inherently unstable system [17], [18]. Further advancements by Abouheaf et al. applied reinforcement learning to pilot control in Ochi's model, demonstrating promise of adaptive control approaches for weight-shift aircraft [19]–[22]. The online reinforcement learning solutions are robust to dynamic model uncertainty that are an issue for model-based control design approaches.

The weight-shift piloting system introduced in this work is realized by a manipulator that employs cable-winch servo motors, forming a cable-driven parallel robot (CDPR) [23]–[25]. CDPRs offer advantages over open-chain manipulators,

This work was supported by Romaeris Corporation and Mitacs under the Accelerate fellowship project *Artificial Intelligence (AI) Powered Adaptive Flight Controller for Novel Unmanned Aerial Vehicle (UAV) with Commercial and Humanitarian Applications.*

including lower weight, higher force to weight ratio, expansive operating workspace, and increased structural stiffness [26]–[28]. The ability of CDPRs to exert large force vectors across any reachable configuration is crucial for WSC applications, matching the capabilities of a human pilot with economical and straightforward hardware configurations [29]–[32]. CDPRs have the inability to transmit compressive forces, complicating control as it requires simultaneous, precise tension adjustments in the cables to achieve the desired end-effector forces.

This work reports on developments of new prototype weight-shift aircraft control designs developed in collaboration with Romaeris Corporation [33]. Previous work has focused on the development of parallel linear actuator designs to replace the human pilot, controlled via model-free reinforcement learning to overcome the parameter uncertainty of the aircraft dynamics [34]. Model-free online reinforcement learning methods were applied in [35] to stabilize the nonlinear dynamics of a surrogate weight-shift mechanism using parallel servo-winch actuators on real flight controller hardware. A full-fledged electronic WSC piloting system based on a CDPR that provides optimal trajectory tracking control was introduced in [36]. The integration of automatic flight controller software with weight-shift aircraft was demonstrated with simulated flights in [37].

II. Experimental flight test

A. Industry partner operations

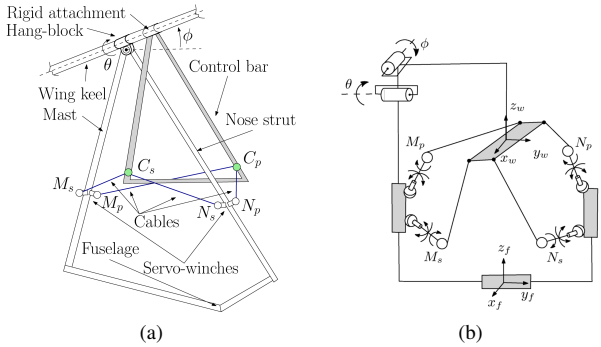
Flight control solutions are designed within predefined constraints on experimental operation for hardware and software. Romaeris' WSC test pilots actively participate in experiment design in order to comply with essential WSC flight principles. All operations maintain stringent adherence to safety and regulatory standards ensuring that experimental flight solutions are safe [2], [38]–[42]. Additionally, robotic prototypes are equipped with emergency stops, and flight control systems have firmware-limited actuation authority to ensure reliability and safety, with all systems undergoing rigorous ground testing prior to flight test.

B. Flight control hardware and instrumentation

An Emlid Navio2 [43] board attached to a Raspberry Pi 3 [44] is selected as the controller's computational unit. Navio2 provides redundant orientation estimates through the combination of sensor readings from either the main onboard InvenSense MPU-9250 9 DOF Inertial Measurement Unit (IMU) [45] or the secondary Microelectronics LSM9DS1 9 DoF IMU [46], as well as global positioning system (GPS). Angular position measurements of the wing orientation are sampled at $20\,\mathrm{Hz}$. The total root mean square (RMS) noise of the MPU-9250 used for inertial measurement feedback is provided to be $0.1\,^\circ/\mathrm{s}$. The measurements data are filtered by a low-pass filter with a cut-off frequency of $250\,\mathrm{Hz}$. The avionics data is captured as onboard data logs.

The open-source unmanned flight controller software framework Ardupilot is selected as the backbone of the onboard flight data collection in a ride-along mode [47]. ArduPilot is not certified or supported for active controls in manned flights. The partner uses proprietary sensors and actuation systems software that control the aircraft with limited authority. ArduPilot providaes a complete set of high-level autonomous control functionality, a full suite of sensor support providing state feedback using a Kalman filter based attitude-heading-reference-system, low-level PID controller loops, support for a multitude of different hardware platforms, and extensive community developer support. Given its open-source nature, the software is easily extensible to new unmanned vehicle types, as shown in [37].

The robotic systems developed followed a progression from mock-ups that mimic the kinematics of WSC aircraft to eventual implementation on WSC aircraft. The experimental platform shown in figure 2a is the most recent iteration of the WSC mockup system. It has identical kinematics to a WSC aircraft prototype. The fuselage is suspended from the wing frame structure by a two DoF mechanical joint that is analogous to the hang-block pivot point of weight-shift aircraft. The control of the platform is actuated by the CDPR. Actuation is provided by four Dynamixel XM-540 servomotors [48] with integrated cable-winches. As illustrated in figure 1, these manipulators are mounted to different positions along the fuselage by spherical joints, and have the other end of their cables attached at positions on the control bar. The servomotors receive torque input commands at a rate of $50\,\mathrm{Hz}$, corresponding to the control update interval $\Delta_t = 20\,\mathrm{ms}$.

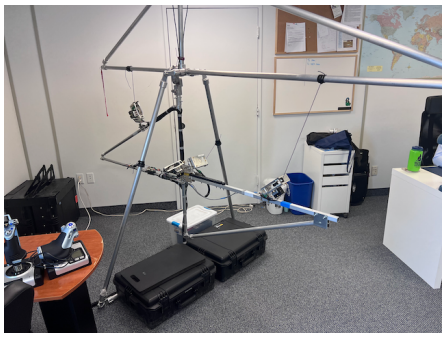


(a) (b)

Fig. 1: (a) Schematic of the wing-fuselage system with attached CDPR piloting system; (b) corresponding kinematic diagram

The CDPR has been implemented on several experimental Romaeris aircraft. Candidate piloting systems were first tested on the converted WSC aircraft shown in figure 2b. The WSC capability was demonstrated by a hanging test where the aircraft was hoisted by the lift point. Reference trajectory tracking was tested, along with more essential validation of the hardware and software functions. Once reliability of the piloting system is proven in physical experiments, a program of experimental flight test with a human pilot is conducted. When active, the CDPR system replaces manual inputs of a

human pilot in supplying the weight-shifting force. The pilot remains in the loop to direct control input.



(a) Experimental WSC mock-up.

(b) CDPR piloting system test.

Fig. 2: Examples of lab experiment hardware employed to test the CDPR piloting system prior to flight test.

(a) Experimental weight-shift aircraft.

(b) Cockpit view of test pilot sharing his evaluation of piloting system's performance with ground crew.

Fig. 3: The CDPR piloting system equipped to weight-shift aircraft. When engaged, the CDPR replaces physical pilot control with electronic FBW control provided by the controller.

C. Experimental weight-shift aircraft

Tests of the experimental weight-shift piloting system were performed with the Cosmos Phase 3 ultralight fuselage equipped with a La Mouette IPSOS 19 flexible wing [49], [50]. The wing is rated for a maximum velocity of $80\,\mathrm{km/h}$, and a total flying weight of $350\,\mathrm{kg}$. An image of the test aircraft is presented in figure 3a. The CDPR actuators are mounted to the chassis and wing down-tubes above the pilot. While this mounting configuration position is sub-optimal in terms of available control authority as the leverage to the hang-block pivot point is reduced, it is an intentional choice that ensures the piloting system out of the way of the human pilot. The servo-motors are further restricted in maximum available torque at the firmware level. Figure 3b shows the pilot operating the FBW piloting system with a leg mounted joystick.

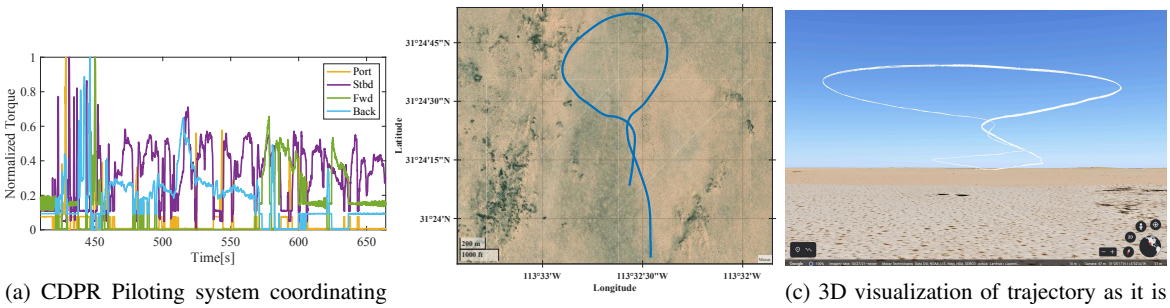
III. Experimental results

A. Sample 1: Maiden test of CDPR piloting system

A test flight program was launched to validate the CDPR's performance in real-life experimental flight scenarios. After reaching safe operating conditions, the test pilot engages the CDPR piloting system and inputs pitch-roll commands via the joystick. The first test result sample is from the maiden test flight of the piloting system, and shows a fixed altitude starboard-side banking turn maneuver that requires coordinated 2-axis control. Plots for the servo-winch torque commands and the corresponding section of the 2D GPS track from the test flight data log are shown in figure 4. During the maneuver the altitude was held in a horizontal plane between $260\,\mathrm{m}$ to $270\,\mathrm{m}$.

B. Sample 2: Automatic flight control with CDPR

The second test sample presents the initial flight test conducted using automatic flight control to manage the attitude of the weight-shift aircraft through the CDPR piloting system. The human pilot remains in the loop, monitoring the flight. In contrast with the first sample, the flight controller is responsible for attitude management, while the human pilot controls throttle and provides desired attitudes. The controller's PID gains and parallel actuator coordination logic are derived from the model-free reinforcement learning solution in [36].



(a) CDPR Piloting system coordinating actuator inputs to induce wing attitude change.

(b) Projected ground track overlaid on satellite map.

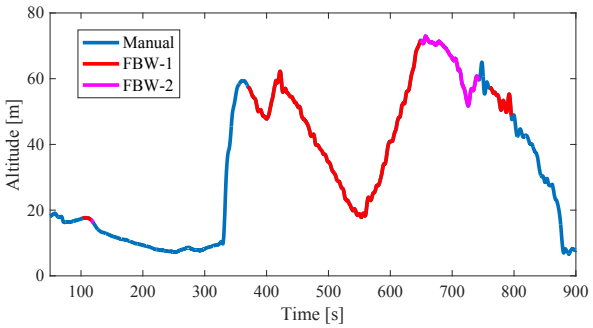
(c) 3D visualization of trajectory as it is seen from south vantage point, rendered within Google Earth.

Fig. 4: Flight trajectory of first sample, with piloting system inputs for coordinated turn maneuver.

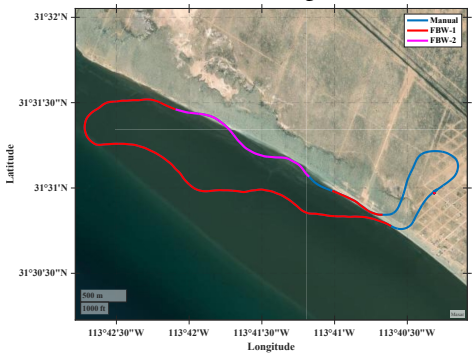
Test pilot notes that average prevailing winds were at $4.7\,\mathrm{m/s}$ with a north-east heading of 54° measured from north. The trajectory data for the sample flight test is presented in figure 5. The flight has three control type phases labelled manual, fly-by-wire mode 1 (FBW-1), and fly-by-wire mode 2 (FBW-2). In manual, the control is a mixture of manual pilot input and direct drive controls with no active feedback control, identical to the first test flight sample. In FBW-1 the feedback controller is commanded by the human pilot to follow roll angles to induce banking turn, pitch is held steady by the CDPR, and throttle is directly controlled by the pilot. FBW-2 is a superset of the FBW-1 mode, where pitch control is additionally employed by the flight controller to manage altitude deviation.

The flight track as shown in figure 5a is better understood when divided into sections as described by the test pilot. The $0\,\mathrm{s}$ to $380\,\mathrm{s}$ section is in manual control as the pilot climbs out from take-off, gains $60\,\mathrm{m}$ of altitude to allow for safe test maneuvers, and performs an in-flight actuation systems check before beginning the automatic controls test. During the $380\,\mathrm{s}$ to $570\,\mathrm{s}$ section, a brief actuation systems check in FBW-1 is performed to ensure positive control is available, after which the pilot allows the aircraft to descend while inputting turn commands as needed to keep the aircraft on its original heading. During the $570\,\mathrm{s}$ to $650\,\mathrm{s}$ section, the pilot recovers altitude, and then performs a single large clockwise turn. During the $650\,\mathrm{s}$ to $750\,\mathrm{s}$ section, the pilot switches to FBW-2 mode to evaluate the pitch-altitude correction response of the controller. The experiment is ended past this time as the pilot switches to manual, disengages the piloting system actuators, and performs landing under human powered input.

In FBW test, the automatic feedback control is interpreting the pilots electronic joystick inputs and correcting for the wing attitude error. Figure 6 shows the data captured of the feedback controller's performance. It can be seen in figure 6a that the inputs are in the positive direction as the human pilot is commanding the clockwise turn corrections to counteract the prevailing wind that is pushing the aircraft away from the original heading. The aircraft enters a $\pm 10^\circ$ dutch roll

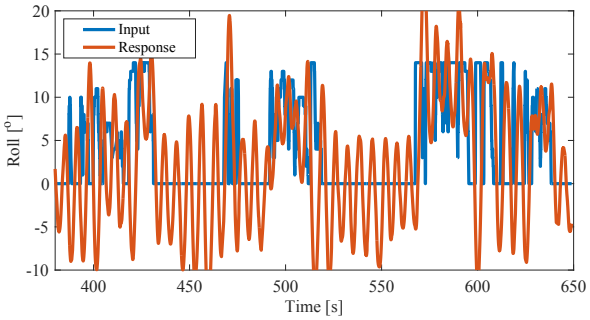


(a) Altitude over time for automatic flight controls test sample.

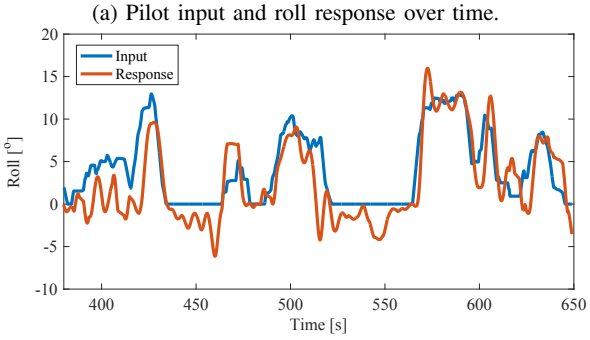


(b) Projected ground track overlaid on satellite map.

Fig. 5: Flight trajectory of the second sample. The automatic flight controls were active for the track sections labelled FBW-1 and FBW-2.

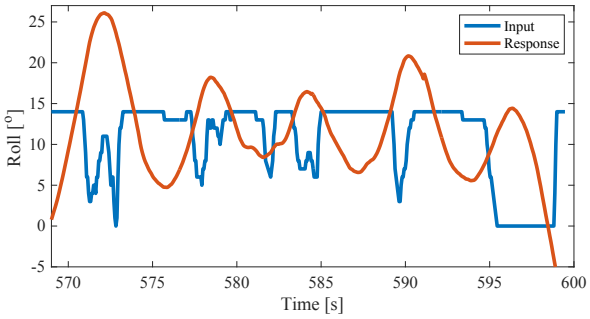


(a) Pilot input and roll response over time.

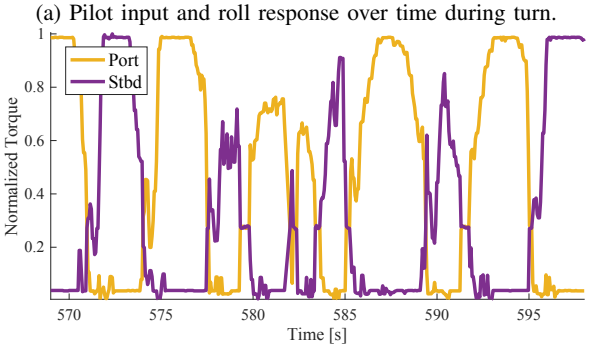


(b) Two second moving averages of pilot input and roll response over time.

Fig. 6: Roll controller input and response of CDPR piloting system while in FBW-1 section of test flight sample.



(a) Pilot input and roll response over time during turn.



(b) The parallel port and starboard side actuators normalized torques over time during the sustained clockwise turn.

Fig. 7: Roll controller input and response of the CDPR piloting system while in FBW-1 for the large sustained clockwise turn.

oscillation due to the control error correction input. The controller could not attenuate the oscillation as it was not adequately tuned and had limited authority. Figure 6b shows the result when a 2 s moving average filter is applied to the input and response data, where it is more readily seen that the control performs adequately despite the feedback oscillation.

The large sustained clockwise turn during the FBW-1 section is worth highlighting for it demonstrates the performance of the piloting system as it operates away from the trim region of the flight envelope. Figure 7 presents the input and response performance of the piloting system during the turn. The piloting system detects the aircraft exceeding the imposed safety limit of 15° and performs an immediate automatic corrective action, overriding the pilot's sustained command. The failsafe action repeats 3 times before the aircraft completes the turn. The automatic action of the parallel actuators corresponding to the clockwise turn is shown in figure 7b. It is clearly observed that the actuators reach the firmware limited saturation point during the turn. This limit prevents the piloting system from performing dangerous weight-shift inputs that have the potential to be catastrophic when near the edge of the flight envelope's stability region.

IV. Conclusion

This work reports on the implementation of a novel cable-driven parallel robot (CDPR) system for automatic weight-shift control in ultralight aircraft. The experimental results validate the piloting system's ability to perform flight maneuvers that would otherwise involve physical input of the human pilot. The PID feedback controllers while able to manage the attitude of the wing were not well-tuned and induced significant oscillations to the system. Future developments will explore the integration of more sophisticated adaptive control techniques to enhance the CDPR's performance in variable flight conditions.

Acknowledgments

We extend our sincere thanks to Bruno Doerwald of Romaeris Corporation for his leadership in orchestrating the real-world flight test campaign.

We are deeply grateful to Romaeris Corporation and Mitacs who provided financial support for this research.

References

[1] G. B. Gratton, "The weightshift-controlled microlight aeroplane," *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, vol. 215, no. 3, pp. 147–154, 2001.

[2] Federal Aviation Administration, *Weight-Shift Control Aircraft Flying Handbook*. Aviation Supplies & Academics, Inc., 2015.

[3] M. J. Kroes and M. S. Nolan, *Weight-Shift Control Aircraft*, 8th ed., ser. Aircraft Basic Science. New York: McGraw-Hill Education, 2013.

[4] E. Geller, "Hang glider stability and control," *Technical Soaring*, vol. 27, no. 1-2, pp. 8–17, 2003.

[5] F. M. Rogallo, "Nasa Research on Flexible Wings," *Annals of the New York Academy of Sciences*, vol. 154, no. 2, pp. 953–961, 1968.

[6] P. Dees, "Hang glider design and performance," in *AIAA Aviation Technology, Integration, and Operations Conference*. Fort Worth, Texas: American Institute of Aeronautics and Astronautics, September 2010.

[7] E. A. Kilkenny, "An experimental study of the longitudinal aerodynamic and static stability characteristics of hang gliders," Ph.D. dissertation, Cranfield Institute of Technology, Sep. 1986.

[8] J. T. Sweeting, "An experimental investigation of hang glider stability," M.Sc. thesis, College of Aeronautics, Cranfield Institute of Technology, 1981.

[9] I. Kroo, "Aerodynamics, aeroelasticity and stability of hang gliders - experimental results," Stanford University and NASA Ames Research Center, Moffett Field, California, technical report NASA-TM-81269, 1981.

[10] E. A. Kilkenny, "An evaluation of a mobile aerodynamic test facility for hang glider wings," College of Aeronautics, Cranfield Institute of Technology, technical report 8330, 1983.

[11] ——, "Full scale wind tunnel tests on hang glider pilots," Cranfield Institute of Technology, College of Aeronautics, Department of Aerodynamics, technical report, 1984.

[12] G. De Matteis, "Response of hang gliders to control," *The Aeronautical Journal*, vol. 94, no. 938, pp. 289–294, 1990.

[13] ——, "Dynamics of hang gliders," *Journal of Guidance Control and Dynamics*, vol. 14, no. 6, pp. 1145–1152, 1991.

[14] W. Khaddage, "A wind-tunnel investigation of an ultra-light wing and ultra-light aircraft," M.A.Sc. thesis, Carleton University, 2017.

[15] M. V. Cook and M. Spottiswoode, "Modelling the flight dynamics of the hang glider," *The Aeronautical Journal*, vol. 109, no. 1102, pp. 1–20, 2005.

[16] M. V. Cook, *Flight Dynamics Principles: A Linear Systems Approach to Aircraft Stability and Control*, 3rd ed., ser. Aerospace Engineering. Butterworth-Heinemann, 2013.

[17] Y. Ochi, "Modeling of the longitudinal dynamics of a hang glider," in *AIAA Modeling and Simulation Technologies Conference*, 2015, pp. 1591–1608.

[18] ——, "Modeling of flight dynamics and pilot's handling of a hang glider," in *AIAA Modeling and Simulation Technologies Conference*, 2017, pp. 1758–1776.

[19] M. Abouheaf, W. Gueaieb, and F. L. Lewis, "Online model-free reinforcement learning for the automatic control of a flexible wing aircraft," *IET Control Theory & Applications*, vol. 14, no. 1, pp. 73–84, 2020.

[20] ——, "Model-Free Gradient-Based Adaptive Learning Controller for an Unmanned Flexible Wing Aircraft," *Robotics*, vol. 7, no. 4, p. 66, Dec. 2018.

[21] M. Abouheaf and W. Gueaieb, "Online model-free controller for flexible wing aircraft: a policy iteration-based reinforcement learning approach," *International Journal of Intelligent Robotics and Applications*, vol. 4, no. 1, pp. 21–43, Mar. 2020.

[22] ——, "Reinforcement learning solution with costate approximation for a flexible wing aircraft," in *IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications*, Jun. 2018, pp. 1–6.

[23] Y. Yiu, H. Cheng, Z. Xiong, G. L. Liu, and Z. Li, "On the dynamics of parallel manipulators," in *Proceedings IEEE International Conference on Robotics and Automation*, vol. 4, 2001, pp. 3766–3771 vol.4.

[24] T. Paty, N. Binaud, S. Caro, and S. Segonds, "Cable-driven parallel robot modelling considering pulley kinematics and cable elasticity," *Mechanism and machine theory*, vol. 159, pp. 104 263–, 2021.

[25] M. H. Vu and U. J. Na, "A new 6-DOF haptic device for teleoperation of 6-DOF serial robots," *IEEE Transactions on Instrumentation and Measurement*, vol. 60, no. 11, pp. 3510–3523, Nov. 2011.

[26] M. Zarebidoki, J. S. Dhupia, and W. Xu, "A review of cable-driven parallel robots: Typical configurations, analysis techniques, and control methods," *IEEE Robotics & Automation Magazine*, vol. 29, no. 3, pp. 89–106, 2022.

[27] I. Ebert-Uphoff and P. A. Voglewede, "On the connections between cable-driven robots, parallel manipulators and grasping," in *Proceedings IEEE International Conference on Robotics and Automation*, vol. 5, 2004, pp. 4521–4526 Vol.5.

[28] H. A. F. Almurib, H. F. Al-Qrimli, and N. Kumar, "A review of application industrial robotic design," in *2011 Ninth International Conference on ICT and Knowledge Engineering*, 2012, pp. 105–112.

[29] H. Khakpour, L. Birglen, and S.-A. Tahan, "Synthesis of differentially driven planar cable parallel manipulators," *IEEE Transactions on Robotics*, vol. 30, no. 3, pp. 619–630, 2014.

[30] F. Pierrot, V. Nabat, O. Company, S. Krut, and P. Poignet, "Optimal design of a 4-dof parallel manipulator: From academia to industry," *IEEE Transactions on Robotics*, vol. 25, no. 2, pp. 213–224, 2009.

[31] N. Mostashiri, J. S. Dhupia, A. W. Verl, and W. Xu, "A review of research aspects of redundantly actuated parallel robotsw for enabling further applications," *IEEE/ASME Transactions on Mechatronics*, vol. 23, no. 3, pp. 1259–1269, 2018.

[32] S. Qian, B. Zi, W.-W. Shang, and Q.-S. Xu, "A review on cable-driven parallel robots," *Chinese journal of mechanical engineering*, vol. 31, no. 1, pp. 29–39, 2018.

[33] "Romaeris Corporation," https://www.romaeris.com, 2024, accessed: April 23 , 2024.

[34] M. Abouheaf, N. Mailhot, and W. Gueaieb, "An Online Reinforcement Learning Wing-Tracking Mechanism for Flexible Wing Aircraft," in *2019 IEEE International Symposium on Robotic and Sensors Environments (ROSE)*, Jun. 2019, pp. 1–7.

[35] M. Abouheaf, N. Q. Mailhot, W. Gueaieb, and D. Spinello, "Guidance Mechanism for Flexible-Wing Aircraft Using Measurement-Interfaced Machine-Learning Platform," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 7, pp. 4637–4648, Jul. 2020.

[36] N. Mailhot, M. Abouheaf, and D. Spinello, "Model-Free Force Control of Cable-Driven Parallel Manipulators for Weight-Shift Aircraft Actuation," *IEEE Transactions on Instrumentation and Measurement*, vol. 73, pp. 1–8, 2024, conference Name: IEEE Transactions on Instrumentation and Measurement.

[37] N. Mailhot, T. de Jesus Krings, G. T. Navajas, B. Zhou, and D. Spinello, "uWSC Aircraft Simulator: A Gazebo-based model for uncrewed weight-shift control aircraft flight simulation," in *2023 IEEE International Symposium on Robotic and Sensors Environments (ROSE)*, Nov. 2023, pp. 1–8.

[38] Transport Canada, "Transport Canada Aeronautical Information Manual (TC AIM) - TP 14371," https://tc.canada.ca/en/aviation/publications/transport-canada-aeronautical-information-manual-tc-aim-tp-14371, Oct. 2022, accessed: March 15, 2023.

[39] "Special Flight Authorizations | Federal Aviation Administration," https://www.faa.gov/aircraft/gen_av/ultralights/sfa, accessed: March 15, 2023.

[40] Transport Canada, "Recreational aircraft airworthiness," 2019, accessed: April 10th, 2024. [Online]. Available: https://tc.canada.ca/en/aviation/aircraft-airworthiness/recreational-aircraft-airworthiness

[41] Agencia Federal de Aviación Civil, "Reglamentos - agencia federal de aviación civil," 2023, accessed: April 10th, 2024. [Online]. Available: https://www.gob.mx/afac/acciones-y-programas/reglamentos-252211

[42] "eCFR - code of federal regulations, title 14, part 103: Ultralight vehicles," 2024, accessed: April 10th, 2024. [Online]. Available: https://www.ecfr.gov/current/title-14/chapter-I/subchapter-F/part-103?toc=1

[43] Emlid Limited, "Emlid official open source documentation," accessed: March 15, 2023. [Online]. Available: https://github.com/emlid/emlid-docs

[44] Raspberry Pi Foundation, "Raspberry Pi official open source documentation," https://github.com/raspberrypi/documentation, accessed: March 15, 2023.

[45] TDK Corp. InvenSense, "TDK InvenSense MPU-9250 sensor documentation," accessed: December 1, 2022. [Online]. Available: https//www.invensense.com/products/motion-tracking-9-axis/mpu-9250/

[46] STMicroelectronics N.V., "STM LSM9DS1 sensor documentation," accessed: January 15th. [Online]. Available: https://www.st.com/en/mems-and-sensors/lsm9ds1.html

[47] ArduPilot Development Team and Community, "ArduPilot open source autopilot," accessed: March 15, 2023. [Online]. Available: http://ardupilot.org/

[48] Robotis Co., "Dynamixel SDK," accessed: March 15, 2023. [Online]. Available: http://emanual.robotis.com/docs/en/software/dynamixel

[49] "Cosmos aerospace overview," http://meiyagroup.com/cosmos-aerospace.html, accessed: July 19, 2023.

[50] "La mouette microlight wings," https://lamouette.com/en/the-ultralight-wing-ranges/, accessed: July 19, 2023.