Routing and Control of Unmanned Aerial Vehicles for Performing Contact-Based Tasks

Robert B. Anderson

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Industrial and Systems Engineering

Andrea L'Afflitto, Chair
Douglas R. Bish
Craig A. Woolsey
Kereshmeh Afsari

April 22, 2021
Blacksburg, Virginia

Keywords: Adaptive Control, Model Reference Adaptive Control, Unmanned Aerial Vehicles, Vehicle Routing Problem

Routing and Control of Unmanned Aerial Vehicles for Performing Contact-Based Tasks

Robert B. Anderson

(ABSTRACT)

In this dissertation, two main topics are explored, the vehicle routing problem (VRP) and model reference adaptive control (MRAC) for unknown nonlinear systems. The VRP and its extension, the split delivery VRP (SVRP), are analyzed to determine the effects of using two different objective functions, the total cost objective, and the last delivery objective. A worst-case analysis suggests that using the SVRP can improve total costs by as much as a factor of 2 and the last delivery by a factor that scales with the number of vehicles over the classical VRP. To test the theoretical worst-cases against the solutions of benchmark datasets, a heuristic is developed based on embedding a random variable neighborhood search within an iterative local search heuristic. Results suggest that the split deliveries do in fact improve total cost and last delivery times over the classical formulation.

The SVRP has been developed classically for use with vehicles such as trucks which have large payload capacities and typically long ranges for deliveries, but are limited to traversing on roads. Unmanned aerial vehicles (UAVs) are useful for their high maneuverability, but suffer from limited capacity for payloads and short ranges. The classical SVRP formulation is extended to one more suitable for UAVs by accounting for limited range, limited payloads, and the ability to swap batteries at known locations. Instead of Euclidean distances, path plans which are adjusted for a known, constant wind underlie the cost matrix of the optimization problem. The effects of payload on the vehicle's range are developed using propeller momentum theory, and simulations verify that the proposed approach could be used in a realistic scenario.

Two novel MRAC laws are then developed. The first, MRAC laws for prescribed performance, exploits barrier Lyapunov functions and a 2-Layer approach to guarantee user-defined performance. This control law allows unknown nonlinear systems to verify a user-defined rate of convergence of the tracking error while verifying apriori control and tracking error constraints. Numerical simulations are performed on the roll dynamics of a delta-wing aircraft. The second novel MRAC law is MRAC for switched dynamical systems which is proven in two different mathematical frameworks. Applying the Carathéodory framework, it is proven that if the switching signal has an arbitrarily small, but non-zero, dwell-time, then solutions of both the trajectory tracking error's and the adaptive gains' dynamics exist, are unique, and are defined almost everywhere, and the trajectory tracking error converges asymptotically to zero. Employing the Filippov framework, it is proven that if the switching signal is Lebesgue integrable and has countably many points of discontinuity, then maximal solutions of both the trajectory tracking error and the adaptive gains dynamics exist and are defined almost everywhere, and the trajectory tracking error converges to zero asymptotically. The proposed MRAC law is experimentally verified in the case where a UAV with tilting propellers is tasked with mounting an unknown camera onto a wall.

The previous results are then combined into a novel application in construction. A method for using a UAV to measure autonomously the moisture of an exterior precast concrete envelope is developed which can provide data feedback through contact-based measurements to improve safety and real-time data acquisition through the integration with the Building Information Model (BIM). To plan the path of the vehicle, the path planning and SVRP for UAV approaches developed in previous chapters are utilized. To enable the UAS to contact surfaces, a switched MRAC law is employed to control the vehicle throughout and guarantee successful measurements. A full physics-based simulation environment is developed, and the proposed framework is used to simulate taking multiple measurements.

Routing and Control of Unmanned Aerial Vehicles for Performing Contact-Based Tasks

Robert B. Anderson

(GENERAL AUDIENCE ABSTRACT)

The main goal of this dissertation is to provide an implementable approach to the routing and control problem for unmanned aerial vehicles (UAVs) tasked with delivering payloads or taking images or videos of known locations. To plan routes for the fleet of vehicles, a split vehicle routing (SVRP) approach is utilized. UAVs are useful for their high maneuverability, but suffer from limited capacity for payloads and short ranges. Before extending the SVRP to a formulation more suitable for UAVs, we study the effects of using two different objective functions on the solutions to the optimization problem through a worst-case analysis. Namely, we study the minimum total cost function and the minimum last delivery function and their effects on both the classical vehicle routing problem (VRP), where only one vehicle can visit each customer, and the SVRP, where multiple vehicles can visit each customer. A custom heuristic is developed to solve several benchmark instances, and the results suggest that using the SVRP can save in total cost and last delivery over the VRP when using the same objective functions.

The classical SVRP formulation is then extended to one more suitable for UAVs by accounting for limited range, limited payloads, and the ability to swap batteries at known locations. Instead of using straight line approaches to traversing between locations, a path planning approach is utilized and wind is accounted for. The effects of payload on the vehicle's range are also considered, and simulations verify that the proposed approach could be used in a realistic scenario.

After developing a routing approach for UAVs, the control problem is considered. The

first control approach developed is for unknown nonlinear systems which necessitate control and tracking error constraints that can be set before the start of the mission. This result is achieved using a novel model reference adaptive control (MRAC) approach. In addition to verifying the constraints, a drawback of classical MRAC approaches, the poor performance in the transient stages, is addressed by providing the ability to guarantee a user-defined rate of convergence of the system. Numerical simulations are performed on the roll dynamics of a delta-wing aircraft.

A second MRAC approach is then developed for the cases in which the UAVs may be tasked with installing a payload at the customer location. An approach is used where the vehicles are considered to have different flight states, one where the vehicle is in free flight, and one where the vehicle contacts the wall. These types of systems are denoted as switched dynamical systems, and an adaptive control law is developed for unknown nonlinear switched plants that must follow the trajectory of user-defined linear switched reference models. The proposed MRAC law is experimentally verified in the case where a UAV with tilting propellers is tasked with mounting an unknown camera onto a wall.

Finally, we seek to combine the new routing and control approach into an application to improve safety within a construction site. A method for using a UAV to measure autonomously the moisture of an exterior precast concrete envelope is developed which can provide data feedback through contact-based measurements to improve safety and real-time data acquisition through the integration with the Building Information Model (BIM). To plan the path of the vehicle, the path planning and SVRP for UAV approaches developed in previous chapters are utilized. To enable the UAS to contact surfaces, a switched MRAC law is employed to control the vehicle throughout and guarantee successful measurements. A full physics-based simulation environment is developed, and the proposed framework is used to simulate taking multiple measurements.

Acknowledgments

First and foremost, I'd like to thank my advisor, Dr. Andrea L'Afflitto. His passion for his work and dedication to his students is invaluable to all of us in his lab. He has mentored me in many academic disciplines from performing research, writing papers, giving presentations, writing proposals, and many others. Perhaps more importantly has been his mentorship outside of research. Several times, I have come to a crossroads in my career, and each time Dr. L'Afflitto has been there to act as both an advisor and a mentor. I wouldn't be where I am today if it wasn't for his guidance.

I would like to thank my committee members Douglas Bish, Kereshmeh Afsari, and Craig Woolsey for taking the time to serve on my dissertation committee, for their challenging comments and discussions, and for reading and improving this work. A special thanks to Dr. Bish and Dr. Afsari for their guidance and expertise in completely new and exciting areas for me.

Paramount to my journey were the three internships with the U.S. Army Research Lab under the guidance of Jim Dotterweich and Harris Edge. They are both the utmost professionals, and the advice and counsel that they provided me throughout those three summers have helped me become the researcher and collaborator that I am today.

This work was supported in part by the US Army Research Lab (ARL), DARPA, and ONR under Grants no. 40304747, D18AP00069, and N000141912422, respectively. I wish to express my gratitude to these sponsors for their continuous support

Contents

List of Figures

List of Tables

List of Acronymns

BIM Building information model

MRAC Model reference adaptive control

RVNS Random variable neighborhood search

SVRP Split vehicle routing problem

TSP Traveling salesman problem

UAV Unmanned aerial vehicle

VRP Vehicle routing problem

Chapter 1

Introduction

Less than 2 years ago, October 18, 2019, Wing made the first commercial drone delivery just miles from the Virginia Tech campus [2]. These unmanned aerial vehicles (UAVs) are becoming more and more accessible and beneficial to the general public by potential expediting short range deliveries. There are several different types of UAVs ranging from multirotors, helicopters, fixed wing, and hybrid vertical takeoff and landing (VTOL), to name a few. The most common type of commercial UAV for the general population today are multirotors, which are typically identified by their number of propellers. For example, a quadrotor as in Figure 1.1a, has 4 motors and propellers, while a hexrotor, see Figure 1.1b, has 6 rotors and propellers. The most useful property of this vehicle type is its capability to vertically take off and land without the need of a runway, and they exhibit the ability to hold position or maneuver in tight spaces. Some weaknesses of multirotor aircraft include low endurance and typically low flight times compared to their fixed-wing counterparts.

Fixed-wing UAVs, such as those in [3], are equipped with wings to generate lift which is much more efficient than employing spinning propellers, and they usually have much longer endurance than multirotor UAVs. The major downside to fixed-wing UAVs is usually the need for a runway or open flat space from which to take off and land. While they typically fly well at higher speeds than multirotors, fixed-wing aircraft usually cannot hover in place or maneuver as well in tight spaces. More recently, hybrid fixed-wing multirotors, have begun to be developed such as those in [4, 5]. These UAVs try to merge the best properties of the other two categroies by using propeller thrust for vertical takeoff, but use wing lift

during horizontal flight to increase flight times. While UAVs vary greatly in size, weight, and capabilities, in this thesis we focus on so-called Class 1 UAVs which have weight, including payloads, of less than 30kgs [6, Ch. 1].

1.1 Motivation

(a) Quadrotor UAV. (b) Hexrotor UAV.

Figure 1.1: Examples of the most common type of multirotor UAVs.

With the increasing variety of UAVs, the applications for them are becoming wide ranging. UAVs were first popularized by military use in applications such as transporting and delivering goods or persons, performing surveillance to gather data or intelligence, and attacking targets [7]. The use cases and interest for military applications are growing, but interest is growing even faster in the civilian market [8]. Some of the most common uses for commercial UAVs include search and rescue operations in outdoor urban areas [9], indoor urban areas [10], and wilderness areas [11], pipeline inspections [12], disaster response [13, 14], forest fire detection and monitoring [15, 16], and construction safety and progress monitoring [17, 18], to name a few.

A future application of UAVs involves utilizing many vehicles in conjunction to accomplish large scale tasks. For example, consider the case of disaster relief wherein there are two different types of UAVs that must visit some areas within the environment. A set of

fixed-wing UAVs can be utilized to quickly reach locations and scan or survey damage and find missing or injured persons. A set of multirotor UAVs can be sent out with supplies such as bandages or water to deliver, and some of these UAVs could be equipped with an arm to mount beacons for establishing communications or targeting an area for rescuers. Such a problem requires optimized route planning for all of the UAVs to complete the mission as quickly as possible as well as advanced nonlinear control algorithms to handle challenges such as sloshing, unknown payloads such as water on the UAVs. Even more challenging is the control problem for the UAVs which may be tasked with installing a sensor.

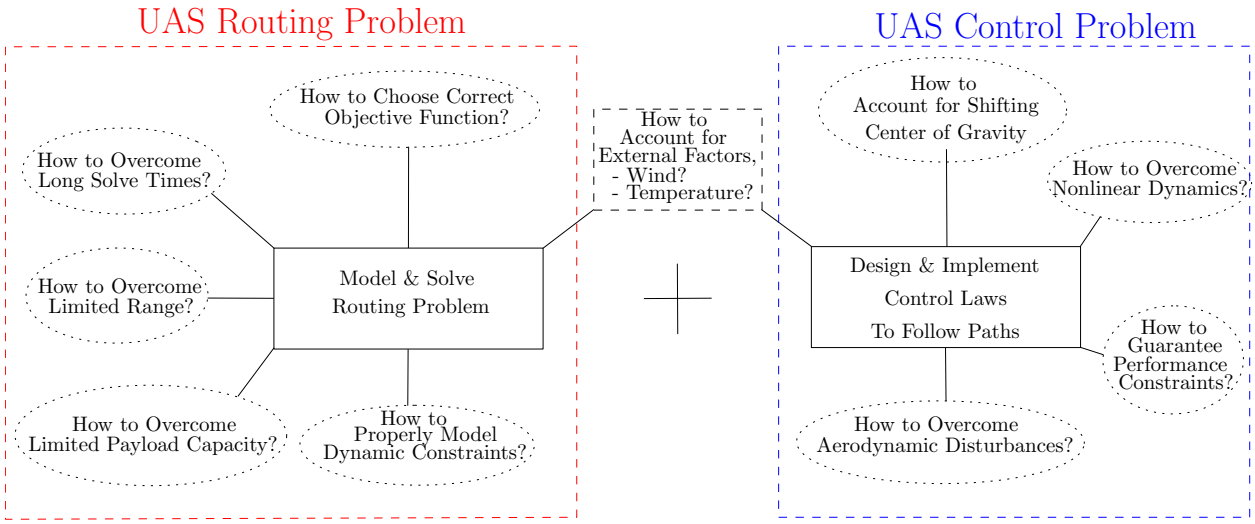

Figure 1.2: Some of the challenging questions involved when solving the routing and control problem for UAVs performing contact-based tasks.

1.2 Goals

In this dissertation, we focus on the problem of using UAVs for delivering payloads to some known delivery locations and performing camera-based or contact-based tasks at these sites. Due to the tasks of both determining the best set of routes for the UAVs to take and the varying challenges of controlling UAVs with unknown payloads flying near hard surfaces, we decompose the problem into two main pieces, the routing problem for UAVs and the control problem for UAVs, and the many of the challenging questions to answer of each are

shown in Figure 1.2. The problem of determining the routes of the UAVs is addressed using a vehicle routing problem (VRP) approach that considers additional characteristics custom to UAVs such as limited flight time, limited payload capacity, endurance dependency on payload, and the payload becomes an optimization variable. In classical vehicle routing problems, a single vehicle is used to service a single customer, or one vehicle per site [19, Ch. 1]. However, for UAVs this may be impractical since these systems tend to have very limited payload capacity. For this reason, we also consider the split vehicle routing problem where multiple vehicles can visit the same location to fill demands. Additionally, since there are a wide range of scenarios where sets of UAVs make deliveries, multiple objective functions are considered. An analysis is performed to show how careful selection of the objective function is vital to the problem at hand.

The nonlinear control problem of UAVs is well-studied in the literature, but it becomes increasingly complex when considering the payload as unknown and unsteady. This modeling choice, while more accurate for unknown moving payloads, can significantly increase the complexity of UAV dynamics [20]. Even further complexity is introduced when considering the UAVs coming into contact with surfaces to perform contact-based tasks, such as camera or beacon installation. Additionally, if the UAVs are being used in proximity to persons or property, it is desirable to have a control system that can an apriori guarantee for the safety of the vehicles and the persons nearby. To overcome the presented challenges, we propose two novel adaptive control laws. The first guarantees safety and high performance in the presence of system unknowns, and the second provides a mechanism for controlling vehicles that can be used for payload installation.

The main goal of this dissertation is to provide a thorough and implementable approach to the routing and control problem for UAVs. Most commonly, research focusing on the routing problem for UAVs ignores both the path planning and control challenges of UAVs, operating under the assumption that the systems are capable of traversing the paths. Paths are commonly considered as straight lines connecting the locations to visit, and while this

can work if flying at a high altitude, considering the routing problem in a 3D environment can exploit the maneuverability benefit of small UAVs. The control problem is typically not considered when planning the optimal routes for large numbers of vehicles in a VRP. For simple missions of visiting locations or carrying precisely known payloads, this can be valid as classical Proportional-Integral-Derivative (PID) autopilots have shown to perform well [21, 22]. The future of UAVs, however, involves performing tasks that interact with the environment and operating in tighter spaces creating much more complicated system dynamics for control purposes.

The overall approach is composed of three main steps. The first part of the proposed approach is to study the effects of choosing different objective functions for the routing problem, which has resulted in a novel study on the split vehicle routing problem. In the second step, a full route optimization approach is developed for UAVs while taking into account some of the characteristics of UAVs that can greatly differ from classical truck routing problems. For example, the cost to go between locations will be generated using a 3D path planner which guarantees complete paths, if feasible, while maintaining a safe distance from other obstacles. These paths are adjusted for a known wind velocity to account for the fact that UAVs fly relative to the air and not the ground. A power model of UAVs, based on momentum theory, is presented and used to capture the effects of carrying extra payloads on the flight times of the vehicles. In the third step, two novel adaptive control laws are developed to ensure that the vehicles can successfully traverse the routes generated by the optimization problem. The first control law guarantees system performance while verifying constraints on the control input. The second control law guarantees stability for systems that have discontinuous dynamics, such as the next generation of UAVs that have to contact surfaces during missions. With all of the pieces of the routing and control problem developed, a novel application is developed wherein a UAV is tasked with taking contact-based autonomous measurements of moisture within a precast concrete surface to improve safety on a construction site.

1.3 Brief outline

In Chapter 2, the objective function of two different VRP formulations is studied both theoretically and numerically through simulations. First, worst-case analysis is performed determine the potential benefits or downsides of utilizing split deliveries. Second, worst-case analysis is performed to examine the effects of using different objective functions on the solution to the optimization problem. Then, a heuristic based on embedding a random variable neighborhood search within an iterative local search heuristic is used to approximate solutions to several benchmark datasets using both the VRP and SVRP formulations as well as both objective functions. We show that using split deliveries can reduce both the total cost and last delivery times, significantly so when the average demand is around 50% of vehicle capacity.

In Chapter 3, a modified SVRP approach is presented to make the routing problem more conducive to UAVs. First, path plans using an A* search algorithm and a known vehicle velocity are used to find the baseline times to travel between nodes. Secondly, these travel times are adjusted to account for a known-constant wind velocity. Thirdly, since UAVs have a range that is dependent on vehicle mass, hence payload, a momentum theory approach is utilized to derive a relation between power and payload mass. This relationship is used as a constraint in the optimization problem to optimize the efficiency of UAV deployment. The optimization model contains not only payload delivery constraints but also the possibility of surveillance time constraints, and since UAVs have a limited flight time, we consider the possibility that the vehicles can stop at known locations to swap batteries. A test-case scenario is provided in which several UAVs must meet demands of 8 customers which includes both payload delivery and surveillance type, multiple refueling stations are available, and the scenario is repeated several times in different wind conditions.

In Chapter 4, a novel model reference adaptive control law for improved transient performance while providing guaranteed saturation constraints on the trajectory tracking error

and the control input at all times is developed.

In Chapter 5, a novel model reference adaptive control law for switched dynamical systems is developed. The control law is first proven in the Carathéodory framework in which any dwell time, or time between state switches, that is strictly positive leads to asymptotic convergence of the trajectory tracking error, and the adaptive gains are bounded. The control law is then proven in the Filippov framework in which asymptotic convergence of the trajectory tracking error is guaranteed, and the adaptive gains are bounded, and the dwell time can be equal to 0. The results are applied to a thrust-vectoring quadrotor whose mission is to mount a sensor on a wall, and the authors believe this is the first experimental validation of an MRAC for switched systems in aerial robotics.

In Chapter 6, we leverage the work in the previous chapters to develop an algorithm that enables a UAV to take contact-based measurements on a construction site which can rapidly update the building information model (BIM). The information from this model is used as the map for an A* path planner which determines the best paths between measurement locations and ensures safe distances from the building. Then, a heuristic solves a TSP problem where the vehicle is allowed to swap batteries and a service time is considered for when the UAV takes measurements. A simulation scenario wherein an actual BIM model is used is presented where a UAV, controlled by the switching MRAC of Chapter 5, takes 2contact wall measurements.

The work presented here is original and has not been submitted previously for a degree, diploma or qualification anywhere else. However, selected parts of this work have been published in the following papers.

1. R. B. Anderson, J. A. Marshall, and A. L'Afflitto. Novel Model Reference Adaptive Control Laws for Improved Transient Dynamics and Guaranteed Saturation Constraints, Journal of the Franklin Institute - Accepted. Key results of Chapter 4.

2. R. B. Anderson, J. A. Marshall, A. L'Afflitto, and J. M. Dotterweich. Model Reference Adaptive Control of Switched Dynamical Systems with Applications to Aerial Robotics, Journal of Intelligent & Robotic Systems - Vol. 100, 3, Nov. 2020, pp. 1265-1281. Key results of Chapter 5.

3. R. B. Anderson, K. Afsari, A. L'Afflitto, S. Halder, and C. Nagori. Designing an Algorithm for Using an Unmanned Aerial System in Contact-based Autonomous Inspection of Building Envelope. Key results of Chapter 6.

Chapter 2

Comparative study of objective functions for classical and split vehicle routing problems

2.1 Introduction

In this dissertation, the problem of deducing the best routes for UAVs in a given scenario is determined using an optimization approach based on the VRP. To this goal, a suitable objective function must be defined. Most commonly, a routing-type problem aims to minimize a cost function based on the total distance traveled or the cost of using the vehicles. This is a solid approach in many large-scale cases, where distance and time can be directly related to cost, and in turn profit, for the users. However, in scenarios such as disaster relief, it may be more important to consider the time at which the last customer is visited, commonly known as the min-max or last delivery routing problem [23]. Since either objective function, or linear combinations thereof, may be appropriate for specific instances, in this chapter both of these common cases are studied. The results are useful later on in determining the appropriate objective function to use in proposed real-world scenarios.

The main purpose of this chapter is to build on the work done in [23] in two key areas. The first point is to determine whether or not using split deliveries is beneficial over the VRP under the same objective function. The second goal is to determine the effects of

using different combinations of objective functions and formulations, and to determine the potential drawbacks of using different objective functions under different lens. The authors studied the traveling salesman problem (TSP) and the VRP under two alternative objectives, minimizing the last delivery, and minimizing the total cost. We study the minimum last delivery objective, and previous results are extended by considering capacity limits on the vehicles and by considering the possibility of split deliveries, denoted the SVRP. One of the main Propositions of the previous paper has since been improved in [24] where the authors show a tighter relation between the last delivery times when using the two different objective functions. However, neither of the previous researchers have considered the relationships between the classical VRP and the SVRP for the different objective functions.

In order to numerically test some of the relationships developed in this paper, a heuristic should be developed to solve benchmark instances for both the VRP and SVRP. Several different heuristic methods have been developed in the literature. Some methods used to solve the VRP with minimum cost objective include tabu search [25], genetic algorithms [26, 27], simulated annealing [28], and ant colony optimization [29], among others. Furthermore, methods such as adaptive variable neighborhood search [30], simulated annealing [31], and adaptive memory [32], among others, have been used to solve the VRP with last delivery objective function. Some recent heuristic methods for solving the SVRP include iterative local search [1], column generation [33], tabu search [34, 35, 36], genetic algorithm [37, 38], among others. The chosen heuristic approach in this work is based on the iterative local search heuristic, which embeds a random variable neighborhood search. This method is chosen as it has been shown to outperform other methods in both the SVRP case [1] and the VRP case for minimum cost objective [39]. Similarly, by slightly modifying the embedded neighborhood search, we show that quick and accurate approximate solutions can also be achieved for the minimum last delivery objective.

The organization of this chapter is as follows. First, in Section 2.2, the notation, the objective functions, and the mixed integer linear programs for both the VRP and the SVRP

are provided and discussed. In Section 2.3, a worst-case analysis is used to compare the VRP and the SVRP under the same objective function to determine the benefits of using split deliveries. In Section 2.4, the two formulations are studied in cases where the objective functions are different with the goal of determining how varying the objective function changes the solutions. In Section 2.5, the potential drawback of using split deliveries for last delivery objective is discussed. Then, the iterative local search heuristic is developed in Section 2.6, and pseudo-algorithms are provided. In Section 2.7, simulations are performed on several benchmark datasets first to validate the usefulness of the heuristic, and the solutions are analyzed to verify the effectiveness of using the SVRP framework.

2.2 Notation, assumptions, and formulations

The problem is defined on a complete graph $G = (V, E)$ with nodes $V = \{0, 1, \ldots, n\}$, where $\{0\}$ denotes the depot or starting location, and n denotes the locations to visit, or customers. For simplicity of notation in the modeling, let N denote the set of customers and the depot, let $N' = N \setminus \{0\}$ denote the set of customers excluding the depot. E denotes the set of edges connecting the vertices, and let $C_{ij} \geq 0$ denote the constant, non-negative travel time to traverse the edge $(i, j) \in E$. It is assumed that the travel times are symmetric, $C_{ij} = C_{ji}$, $(i, j), (j, i) \in E$, and the travel times satisfy the triangle inequality, that is, $C_{ij} \leq C_{ik} + C_{kj}$, $\forall (i, j),\ (i, k),\ (k, j) \in E$. To service the customers, there are n_{veh} homogeneous vehicles with capacity $q > 0$. Each customer in N' has a demand denoted $d_i \in [0, q]$, where in this formulation we are considering cases where customer demand never exceeds the capacity of a single vehicle. This allows for VRP formulations to be tested on all of the networks since no more than 1 vehicle is required to visit each node. Furthermore, an assumption used in this work is that all vehicles must visit at least one customer, and we require that $n_{\text{veh}} \leq n - 1$, or that there is at most 1 vehicle for each node not including the depot.

Remark 2.1. Some of the above assumptions are restrictive when considering applications to routing for UAVs. Travel costs may be asymmetric in cases where there is significant wind in the area. Since UAVs travel with velocity with respect to the air and not the ground, it can occur where traveling one direction between two nodes could be much faster than the converse. Additionally, in many UAV routing instances, more than one vehicle may need to visit a location just to satisfy the demand simply due to the very limited payload capacities of the vehicles. These assumptions are helpful in deducing some of the bounds for the worst-case analysis in this chapter when studying objective functions, but we relax these assumptions in chapter 3 to introduce a formulation more conducive to UAV routing.

A mixed integer programming formulation is utilized. The main decision variable, denoted by x_{ij}^k, is a binary variable that is equal to 1 if vehicle k travels along the (i, j) arc, and is 0 otherwise. The decision variable regarding deliveries is given by $y_i^k \geq 0$ which is the fraction of demand at node $i \in N'$ that is satisfied by vehicle k. Objective functions are typically a combination of either total travel times by the vehicles or the times at which nodes are visited by the vehicles. The three objective functions most commonly used in the VRP or SVRP framework are the minimum cost objective, z_c, the *total cost* objective or the total sum of travel times or distances, the *last delivery* objective z_{ld}, which is the largest time at which a node is visited by a vehicle, typically measured in time, and the *average delivery* objective, z_{ad}, typically measured in time. The total cost objective,

$$z_c \triangleq \sum_{k=1}^{m} \sum_{(i,j)\in E} C_{ij} x_{ij}^k, \tag{2.1}$$

is commonly utilized in a cost optimization scenario where the total mileage or time is minimized to reduce costs. The last delivery,

$$z_{ld} \triangleq \max_{k\in\{1,\ldots,n_{\text{veh}}\}} \left[\sum_{(i,j)\in E: j\neq 0} C_{ij} x_{ij}^k \right], \tag{2.2}$$

can be utilized for disaster relief scenarios or those in which the consideration of the customer waiting the longest is paramount. It is noted that in the case of split deliveries, the demand is considered as satisfied when the entire demand is filled and not when reached by the first vehicle. The average delivery time,

$$z_{ad} \triangleq \frac{1}{m} \sum_{k=1}^{m} \sum_{(i,j)\in E: j\neq 0} C_{ij} x_{ij}^{k}, \tag{2.3}$$

can be used when considering customer satisfaction in deliveries by trying to minimize the average time that each customer has to wait. The objective function in the overall formulation is given as a linear combination of these three possibilities for reasons that are discussed at the end of the next section. The optimal solution is always denoted as $(\cdot)^*$, for example the best minimum cost solution would be denoted as z_c^*.

The mixed integer formulation is given by,

$$\min \quad \lambda_1 z_c + \lambda_2 z_{ld} + \lambda_3 z_{ad}, \tag{2.4}$$

$$\text{s.t.} \sum_{i:(i,j)\in E} x_{ij}^{k} = \sum_{i:(j,i)\in E} x_{ji}^{k}, \quad \forall j \in N, \qquad k \in \{1,\ldots,n_{\text{veh}}\}, \tag{2.5}$$

$$\sum_{(0,j)\in E} x_{0j}^{k} = 1, \qquad k \in \{1,\ldots,n_{\text{veh}}\}, \tag{2.6}$$

$$\sum_{k=1}^{n_{\text{veh}}} \sum_{i\in N} x_{ij}^{k} \geq 1, \qquad j \in N', \tag{2.7}$$

$$y_j^k \leq \sum_{i:(i,j)\in E} x_{ij}^{k}, \qquad j \in N', \qquad k \in \{1,\ldots,n_{\text{veh}}\}, \tag{2.8}$$

$$\sum_{k=1}^{n_{\text{veh}}} y_j^k = 1, \qquad j \in N', \tag{2.9}$$

$$\sum_{j\in N'} d_j y_j^k \leq q, \qquad k \in \{1,\ldots,n_{\text{veh}}\}, \tag{2.10}$$

$$\sum_{i\in S} \sum_{j\in S} x_{ij}^{k} \leq |S| - 1, \qquad S \subset N : |S| \geq 2, \qquad k \in \{1,\ldots,n_{\text{veh}}\}, \tag{2.11}$$

$$z_{ld} \geq \sum_{(i,j)\in E: j\neq 0} C_{ij} x_{ij}^{k}, \qquad k \in \{1, \ldots, n_{\text{veh}}\}. \tag{2.12}$$

The objective function (2.4), where $\lambda_1, \lambda_2, \lambda_3 > 0$, captures a linear combination of the objective metrics, (2.1), (2.2), and (2.3). These metrics are analyzed as stand alone objectives, but it will also be discussed when various combined objectives are useful. Constraint (2.5) is the flow conservation constraint for each vehicle to guarantee the number of arcs flowing into a node equals the number of arcs leaving the node. This ensures that a vehicle must arrive at a node to be able to depart from it. Constraint (2.6) ensures that all vehicles leave the depot. Constraint (2.7) requires all delivery nodes to be visited, potentially by multiple vehicles, thus allowing split deliveries. By changing (2.7) from an inequality to an equality, this becomes a VRP formulation wherein a maximum of 1 vehicle can visit a given node. Constraint (2.8) requires a vehicle to visit a node to satisfy any of the node's demand. Constraint (2.9) requires that each delivery node $i \in N'$ have its demand satisfied, while (2.10) enforces each vehicles capacity limitation. Constraint (2.11) eliminates infeasible sub-tours. Finally, constraint (2.12) captures the last delivery.

In the next sections, worst-case analysis is performed on some of the potential combinations of formulations and objective functions. The analysis is performed in two key steps. First, a bound is proposed between two different combinations of objective functions and formulations. Second, an example is provided to show that this bound is tight. These bounds and examples are further broken down into two different groups. The first group consists of bounds that compare the different formulations while using the same objective functions. The results show the potential benefits of using the SVRP over the VRP. Secondly, analysis is performed in the cases that different objective functions are used, and the key result is that the chosen objective function plays a major role in shaping the solutions.

The notation $metric(model_{objective})$ is utilized to denote the objective function metric evaluated on which formulation was solved using a specified objective function. For example,

to specify the last delivery (ld) from VRP that minimizes last delivery (ld) objective as $ld(VRP_{ld})$, while we specify the cost (c) from a VRP that minimizes last delivery objective and allows split delivery as $c(SVRP_{ld})$. Note that a $(\cdot)^*$ is used to denote an optimal solution. For example, z^*_{ld} denotes the optimal solution of $ld(VRP_{ld})$.

2.3 Evaluation of formulations when using the same cost function

In this section, the potential benefits of using the SVRP over the VRP are shown when both formulations use the same objective function. Before discussing the relationships between the VRP and the SVRP, the relation between the different objective functions is shown.

Proposition 2.2. *For any feasible solution to VRP or $SVRP$, it holds that the average delivery time is less than or equal to the last delivery time, which is strictly less than the total cost,*

$$z_{ad} \leq z_{ld} < z_c. \tag{2.13}$$

Proof: Since z_{ad} is the average of the last delivery times, and z_{ld} is the largest last delivery, $z_{ad} \leq z_{ld}$ must hold, and $z_{ad} = z_{ld}$ only if the last delivery times for all vehicles are equal. The largest last delivery time only includes the time to get to the last node on the route, and since this does not include the return trip, the largest last delivery must be strictly less than the total cost, z_c, which includes all arcs. ■

The next proposition shows that using split deliveries can greatly reduce cost over the VRP when using minimum cost objective.

Proposition 2.3 ([40])**.** *Using the SVRP formulation can improve the total cost by at most*

a factor of 2 over using the VRP,

$$c(VRP_c) \leq 2\ c(SVRP_c). \tag{2.14}$$

The proof relies on giving all nodes a demand that is slightly above half of the vehicle demand. In this case, the VRP must send a vehicle on an out-and-back tour to every node, whereas the SVRP can utilize splits to service 3 nodes for every 2 vehicles to reduce cost. In addition to improving the total cost objective, the next proposition shows that split deliveries can also be quite effective at minimizing the last delivery. First, recall the following result.

Proposition 2.4 ([41])**.** *The last delivery objective for n_{veh} vehicles is at most $2n_{\text{veh}} - 1$ times better than the last delivery for a single TSP tour,*

$$\frac{ld(TSP_{ld})}{ld(VRP_{ld})} \leq 2n_{\text{veh}} - 1. \tag{2.15}$$

The proof for this proposition uses the idea that the VRP will utilize all of the n_{veh} available vehicles. By then letting the last delivery of all of the n_{veh} tours be the maximum last delivery time, the result follows. A similar logic can be used to extend this result to the SVRP as shown in the following proposition.

Proposition 2.5. *The last delivery objective for the SVRP is up to $2n_{\text{veh}} - 1$ times better than the last delivery for when optimizing the VRP for last delivery,*

$$\frac{ld(VRP_{ld})}{ld(SVRP_{ld})} \leq 2n_{\text{veh}} - 1. \tag{2.16}$$

Proof: This proof is made up of a generalized example. Given a set of n nodes, we have n_1 nodes far away from the depot, and $n_2 = n_1 - 1$ nodes arbitrarily close to the depot, with $n = n_1 + n_2$. There are $n_{\text{veh}} = n - 2$ vehicles with capacity q. The demands are as follows, all nodes in n_1 are of demand 1, and all nodes in n_2 have demand equal to vehicle capacity.

In this case, the VRP is forced to a solution with out-and-back tours to the nodes close to the depot, and 1 TSP-like tour for the n_1 nodes. The total solution in the limit of long arcs behaves like the TSP solution in Proposition 2.4. For the split delivery, 1 vehicle can go to all n_2 nodes and make 1 unit delivery. Then, the rest of the vehicles each fulfill the delivery of 1 node close to the depot and one node far from the depot, thus mimicking behavior of the VRP solution from Proposition 2.4. By reducing the SVRP case to the VRP in the previous instance, and the VRP to the TSP in the previous instance, the result follows.

In the following, we provide an example to illustrate the relevance of the proposed results. For all examples in this chapter, the depot is a square with label 0, and the customers are circles with corresponding number labels, see Figure 2.2. Available arcs are shown by lines connecting the different nodes, and the costs are listed. In cases that demands are uniform for customers, the demands are left off of the figures. If the demands are nonuniform, the demands are listed next to the customers.

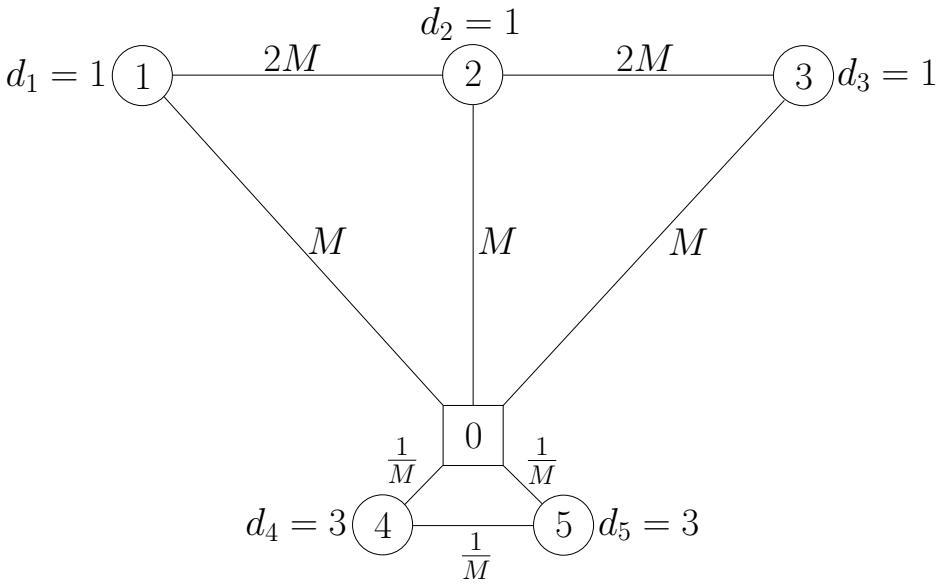

Figure 2.1: Example of tight worst case bound for Proposition 2.5.

The example shows that the upper bound provided by Proposition 2.5 can be tight. Consider the network in Figure 2.1, where the demands are $d_2 = d_3 = d_4 = 1$ and $d_5 = d_6 = 3$, the number of vehicles is $n_{\text{veh}} = 3$, and the vehicle capacity is $q = 3$. The VRP_{ld} has the

optimal tours given in Table 2.1. The total cost is given by $c(VRP_{ld}) = (M + 2M +$

Table 2.1: Optimal routes and associated costs for the VRP.

Vehicle	Route	Cost
1	$0 \to 1 \to 2 \to 3 \to 0$	$M + 2M + 2M + M$
2	$0 \to 4 \to 0$	$1/M + 1/M$
3	$0 \to 5 \to 0$	$1/M + 1/M$

$2M + M) + (1/M + 1/M) + (1/M + 1/M) = 6M + 4/M$. The last delivery is found as $ld(VRP_{ld}) = \max((M + 2M + 2M), (1/M), (1/M) = 5M$. The $SVRP_{ld}$ has the optimal tours given in Table 2.2. where this method exploits the split deliveries to avoid the longer

Table 2.2: Optimal routes and associated costs for the SVRP.

Vehicle	Route	Cost
1	$0 \to 4 \to 1 \to 0$	$1/M + (M + 1/M) + M$
2	$0 \to 5 \to 2 \to 0$	$1/M + (M + 1/M) + M$
3	$0 \to 4 \to 5 \to 3 \to 0$	$1/M + 1/M + (M + 1/M) + M$

TSP-like tour that contains the longest segments, from $1 \to 2 \to 3$. The total cost is given by $c(SVRP_{ld}) = (1/M + (M + 1/M) + M) + (1/M + (M + 1/M) + M) + (1/M + 1/M + (M + 1/M) + M) = 6M + 7/M$. The last delivery is found as $ld(SVRP_{ld}) = \max((1/M + (M + 1/M)), (1/M + (M + 1/M)), (1/M + 1/M + (M + 1/M))) = M + 3/M$. The ratio of the two last delivery metrics is given by

$$\frac{ld(VRP_{ld})}{ld(SVRP_{ld})} = \frac{5M}{M + 3/M} = \lim_{M \to \infty} \frac{5M}{M + 3/M} \to 5 = 2n_{\text{veh}} - 1.$$

△

In this section, it has been shown that utilizing split deliveries can potentially improve the total cost by a factor of 2 and asymptotically improve the last delivery as the number of vehicles increases.

2.4 Evaluating the same metric while optimizing different cost functions

One of the main points of this chapter is to investigate how using one objective function can affect an evaluation of the solution using another metric. For example, it is intuitive to think that minimizing the last delivery improves the likelihood of saving everyone in a disaster scenario, but it is desirable to know what additional cost may be incurred by using the last delivery objective over the minimum cost. Similarly, it is desirable to know how much the last delivery can increase by when using the total cost objective. To this goal, the VRP is evaluated using the last delivery metric when using both the minimum cost objective and the minimum largest last delivery objective.

Proposition 2.6. *The last delivery of the VRP using minimum cost objective is at worst a factor of* $2n_{\text{veh}}$ *larger than optimizing for minimum largest last delivery.*

$$ld(VRP_c) \leq 2n_{\text{veh}}\ ld(VRP_{ld}). \tag{2.17}$$

Proof: For the optimal $ld(VRP_{ld})$ solution, the time of each vehicle's last delivery is denoted as $a_i,\ i = 1, \ldots, n_{\text{veh}}$. Since the largest last delivery $ld(VRP_{ld}) \triangleq \max_{i\in\{1,\ldots,n_{\text{veh}}\}} a_i$, we have that $a_i \leq ld(VRP_{ld})$, $i = 1, \ldots, n_{\text{veh}}$, which yields

$$\sum_{i=1}^{n_{\text{veh}}} a_i \leq n_{\text{veh}}\ ld(VRP_{ld}). \tag{2.18}$$

By the triangle inequality, we have that the return trip to the depot is at most the last delivery time, a_i, and we have that

$$c(VRP_{ld}) \leq 2\sum_{i=1}^{n_{\text{veh}}} a_i \leq 2n_{\text{veh}} \max_{i\in\{1,\ldots,n_{\text{veh}}\}} a_i = 2n_{\text{veh}}\ ld(VRP_{ld}). \tag{2.19}$$

By definition $c(VRP_c) \leq c(VRP_{ld})$ since $c(VRP_c)$ is the optimal minimum total cost. Combining the two results gives,

$$c(VRP_c) \leq c(VRP_{ld}) \leq 2\sum_{i=1}^{m} a_i, \tag{2.20}$$

and by Proposition 2.2

$$ld(VRP_c) \leq c(VRP_c). \tag{2.21}$$

By combining (2.19), (2.20), and (2.21), we have,

$$ld(VRP_c) \leq 2n_{\text{veh}}\ ld(VRP_{ld}). \tag{2.22}$$

■

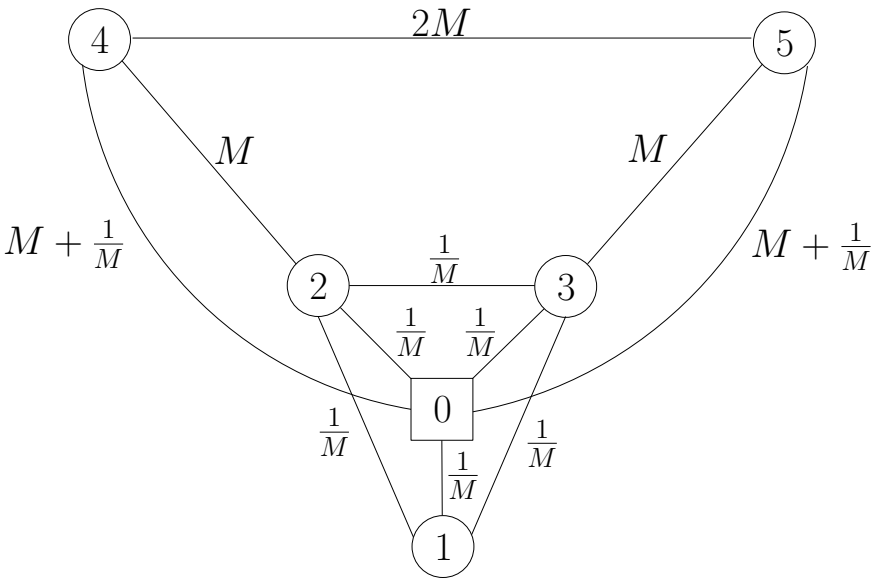

Figure 2.2: Example of tight worst case bound for Proposition 2.6. All locations have a demand of 1.

The following example shows that the upper bound of Proposition 2.6 can be tight. Consider two vehicles, $n_{\text{veh}} = 2$, of capacity four, $q = 4$, and the network in Figure 2.2 where $n = 6$ and $d_i = 1,\ i = 1, \cdots, 5$. The optimal tours for the VRP using minimum

cost objective are provided in Table 2.3. The total cost is given by summing the costs of

Table 2.3: Optimal routes and associated costs under total cost objective.

Vehicle	Route	Cost
1	$0 \to 2 \to 4 \to 5 \to 3 \to 0$	$1/M + M + 2M + M + 1/M$
2	$0 \to 1 \to 0$	$1/M + 1/M$

each vehicle, $c(VRP_c) = (1/M + M + 2M + M + 1/M) + (1/M + 1/M) = 4M + 4/M$. The last delivery is found as the maximum of the costs excluding the return to the depot, $ld(VRP_c) = \max(1/M + M + 2M + M, 1/M) = 4M + 1/M$. The optimal tours for the VRP using last delivery objective are given in Table 2.4. The total cost is given by $c(VRP_{ld}) =$

Table 2.4: Optimal routes and associated costs under last delivery objective.

Vehicle	Route	Cost
1	$0 \to 2 \to 4 \to 0$	$1/M + M + (M + 1/M)$
2	$0 \to 1 \to 3 \to 5 \to 0$	$1/M + 1/M + M + (M + 1/M)$

$(1/M+M+M+1/M)+(1/M+1/M+M+M+1/M) = 4M+5/M$. The last delivery is found as, $ld(VRP_{ld}) = \max(1/M + M + (M + 1/M), 1/M + 1/M + M + (M + 1/M)) = 2M + 3/M$. The ratio of the two last deliveries is given by

$$\frac{ld(VRP_c)}{ld(VRP_{ld})} = \frac{4M + \frac{1}{M}}{M + \frac{2}{M}} = \lim_{M \to \infty} \frac{4M + \frac{1}{M}}{M + \frac{2}{M}} \to 4 = 2n_{\text{veh}}.$$

△

Observation 1. According to Proposition 2.6, the worst-case bounds only rely on a vehicle capacity that is sufficiently large. If we consider the optimal TSP tour on a network, where with large enough capacity 1 vehicle can serve the longest optimal TSP tour, minimizing cost, then we can continue to create these scenarios.

This observation can be verified by looking at the example for Proposition 2.5, where one vehicle does a TSP-like tour and services all customers outside of those near the depot, which in the limit do not contribute to the cost. In [41], a similar bound is proposed, but

in that paper, it is not required to use all vehicles. Proposition 4 of [41] can be extended to our Proposition 2.6 with Observation 1, wherein 1 vehicle services all nodes away from the depot, and additional nodes are added arbitrarily close to the depot to "absorb" the rest of the vehicles, essentially making the result appear as a TSP problem. The next proposition extends the result to the case of split deliveries.

The following proposition analyzes the VRP case for the two different objective functions.

Proposition 2.7. *The cost of the VRP using minimum largest last delivery objective is at worst* n_{veh} *times worse than the cost of the VRP which minimizes cost,*

$$c(VRP_{ld}) \leq n_{\text{veh}}\ c(VRP_c). \tag{2.23}$$

Proof: Directly from [23] Proposition 5, which states,

$$c(TSP_{ld}^{n_{\text{veh}}}) \leq n_{\text{veh}}\ c(VRP_c), \tag{2.24}$$

where $c(TSP_{ld}^{n_{\text{veh}}})$ stands for the cost metric of a TSP with m vehicles that is minimizing last delivery. Since we argue all vehicles must be used, $c(VRP_{ld}) = c(TSP_{ld}^{n_{\text{veh}}})$, because when optimizing the last delivery either the VRP or TSP will use all vehicles to minimize the longest route. ■

The following example shows that the upper bound of Proposition 2.7can be tight. Consider the example in Figure 2.3. There are three customers all with demand 1, $n = 4$. There are 2 vehicles, $n_{\text{veh}} = 2$, with capacity of 2, $q = 2$. The optimal tours for the VRP_{ld} are given in Table 2.5. The total cost is given by $c(VRP_{ld}) = (M + M) + (1/M + (M + 1/M) + M) =$

Table 2.5: Optimal routes and associated costs under last delivery objective.

Vehicle	Route	Cost
1	$0 \to 1 \to 0$	$M + M$
2	$0 \to 3 \to 2 \to 0$	$1/M + (M + 1/M) + M$

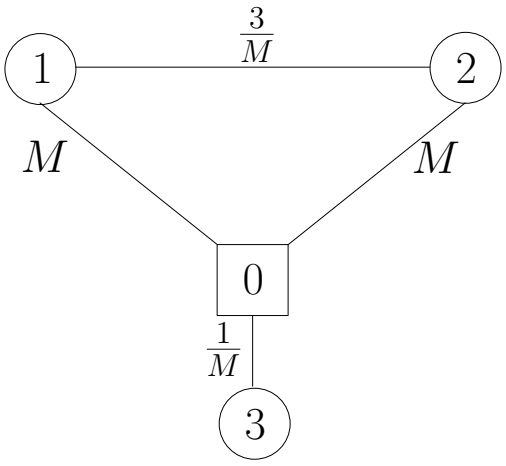

Figure 2.3: Example of tight worst case bound for Proposition 2.7.

$4M + 2/M$. The last delivery is found as $ld(VRP_{ld}) = \max((M), (1/M + (M + 1/M))) = M + 2/M$. The VRP_c has the optimal tours given in Table 2.6. The total cost is given by

Table 2.6: Optimal routes and associated costs under total cost objective.

Vehicle	Route	Cost
1	$0 \to 1 \to 2 \to 0$	$M + 3/M + M$
2	$0 \to 3 \to 0$	$1/M + 1/M$

$c(VRP_{ld}) = (M + 3/M + M) + (1/M + 1/M) = 2M + 5/M$. The last delivery is found as $ld(VRP_{ld}) = \max((M + 3/M), (1/M)) = M + 3/M$. The ratio of the cost metrics using the two different objective functions is given by

$$\frac{c(VRP_{ld})}{c(VRP_c)} = \frac{4M + 5/M}{2M + 5/M} = \lim_{M \to \infty} \frac{4M + 5/M}{2M + 5/M} \to 2 = n_{\text{veh}}.$$

△

In this section, it has been shown that choosing one objective function can have a significant impact on another solution metric. It is shown in Proposition 2.6 that for the VRP the last delivery can potentially increase by a factor of $2n_{\text{veh}}$ when using the total cost objective which could be a very poor choice in emergency scenarios. Furthermore, it is shown in Proposition 2.7 that the VRP with last delivery objective can potentially cost n_{veh} times

more that simply minimizing total cost.

2.5 Potential drawback of using split deliveries

In Section 2.3, it is shown that using the split delivery formulation can only improve the total cost or last delivery objectives when compared to the VRP. However, a potential downside to the SVRP is that when optimizing the last delivery, the total cost increases with a rate equal to the number of vehicles.

Proposition 2.8. *The cost of minimizing the last delivery of the SVRP becomes asymptotically worse than the VRP under last delivery objective,*

$$c(SVRP_{ld}) \leq n_{\text{veh}}\ c(VRP_{ld}). \tag{2.25}$$

Proof: First, we have that

$$\frac{c(VRP_c)}{c(SVRP_{ld})} \leq \frac{c(VRP_{ld})}{c(SVRP_{ld})},$$

since the best scenario for the VRP is to minimize the total cost. The best case for $c(VRP_c)$ occurs when one vehicle does a TSP-like tour, and the other vehicles service a node arbitrarily close to the depot. The worse case for $c(SVRP_{ld})$ is when all vehicles do an out-and-back, or visit one node and return to the depot. This is the structure of optimal solutions to VRP_{ld}. This suggests the worst-case scenario is when $c(SVRP_{ld}) = c(VRP_{ld})$, so we have

$$\frac{c(VRP_c)}{c(VRP_{ld})} = \frac{c(VRP_c)}{c(SVRP_{ld})} \leq \frac{c(VRP_{ld})}{c(SVRP_{ld})} \tag{2.26}$$

and by applying Proposition 2.7,

$$\frac{1}{n_{\text{veh}}} \leq \frac{c(VRP_c)}{c(VRP_{ld})} = \frac{c(VRP_c)}{c(SVRP_{ld})} \leq \frac{c(VRP_{ld})}{c(SVRP_{ld})}, \tag{2.27}$$

$$\frac{1}{n_{\text{veh}}} \leq \frac{c(VRP_{ld})}{c(SVRP_{ld})}, \tag{2.28}$$

which verifies the result. This shows that in the limit as n_{veh} gets arbitrarily large, the SVRP can cost arbitrarily more than the VRP. ■

For an example of the minimum of this ratio, we need to minimize the VRP cost while maximizing the SVRP cost. This will happen when the split delivery can decrease last delivery to far away nodes by exploiting splits, but then for each split we add another "long tour."

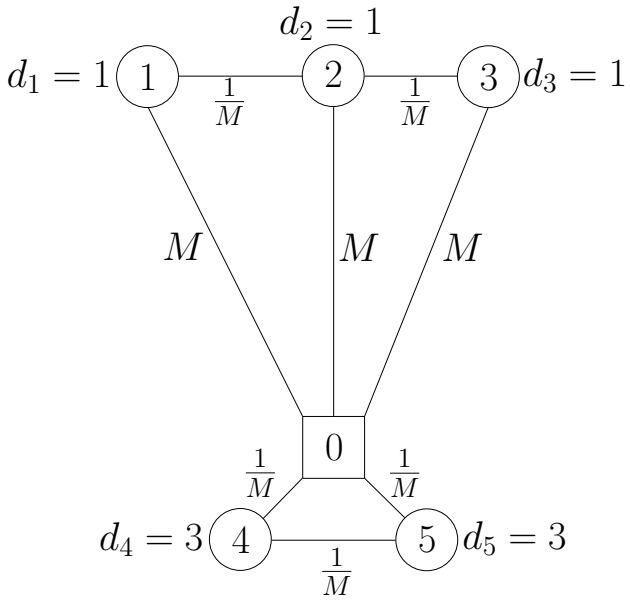

Figure 2.4: Example of lower bound of Proposition 2.8.

This example shows that the lower bound in Proposition 2.8 can be tight. Consider the following example, see Figure 2.4, where there are 6 nodes, the number of vehicles is $n_{\text{veh}} = 3$, and the vehicle capacity is $q = 3$. The VRP_{ld} has optimal routes, note this is the only solution and is designed this way, given in Table 2.7. The total cost is given by $c(VRP_{ld}) = (1/M + 1/M) + (1/M + 1/M) + (M + 2/M + 2/M + M) = 2m + 8/M$. The last

Table 2.7: Optimal routes and associated costs for the VRP.

Vehicle	Route	Cost
1	$0 \to 4 \to 0$	$1/M + 1/M$
2	$0 \to 5 \to 0$	$1/M + 1/M$
3	$0 \to 1 \to 2 \to 3 \to 0$	$M + 2/M + 2/M + M$

delivery is found as $ld(VRP_{ld}) = \max(1/M, 1/M, M+4/M) = M+4/M$. For the SVRP, we force longer tours by using vehicles to partially fill the 2 nodes closest to the depot. From here, the vehicles all perform a longer tour similar to the single longest tour of the VRP. The optimal routes are given in Table 2.8. The total cost is given by $c(SVRP_{ld}) = ((1/M)+(M+$

Table 2.8: Optimal routes and associated costs for the SVRP.

Vehicle	Route	Cost
1	$0 \to 4 \to 1 \to 0$	$(1/M) + (M + 1/M) + M$
2	$0 \to 5 \to 3 \to 0$	$(1/M) + (M + 1/M) + M$
3	$0 \to 4 \to 5 \to 2 \to 0$	$(1/M) + (1/M) + (M + 1/M) + M$

$1/M)+M)+((1/M)+(M+1/M)+M)+((1/M)+(1/M)+(M+1/M)+M) = 6M+7/M$. The last delivery is found as $ld(VRP_{ld}) = \max(M+2/M, M+2/M, M+3/M) = M+3/M$. In this case, the SVRP finds a solution that is marginally better than the VRP, $M+3/M < M+4/M$, but the ratio of cost is given by,

$$\frac{c(VRP^4_{ld})}{c(SVRP^4_{ld})} = \frac{2M+8/M}{6M+7/M} = \lim_{M\to\infty} \frac{2M+8/M}{6M+7/M} \to \frac{1}{3} = \frac{1}{n_{\text{veh}}}.$$

△

The above example easily scales with the number of vehicles as for each additional vehicle, a new node is added close to the depot, and a new node is added far away. The demands are adjusted such that the nodes close to the depot have a demand of $d = q$, and the nodes far away from the depot all have a demand of 1. The routes then proceed in the same manner as the above example. This is an interesting result that suggests that the SVRP could cost infinitely more in the limit than the VRP when minimizing last delivery. This is one of the reasons for including the ability to optimize a linear combinations of the chosen

objective functions. By carefully selecting λ_1 and λ_3, the solution of the SVRP can be the same as the VRP, where the optimal last delivery is not achieved, but there is a balance between the two objective functions. Another example of why it may be useful to have the linear combination of objective functions is that the last delivery objective function has many optimal solutions in many cases. It can be seen by the triangle inequality that any time the optimal last delivery is equal to $\max_{i \in N'} c_{0i}$, then the optimal is achieved. There may be infinitely many ways to serve the rest of the customers while retaining that out-and-back tour for the best last delivery. In this case, it may be useful to let $0 < \lambda_1 << 1$ where the cost function contributes a very small amount to the objective function, but it helps break the ties between the possibly many optimal last delivery solutions. It can easily be seen that as long as $\lambda_1 z_c$ is smaller than any combination of tours that is greater than z^*_{ld}, then the optimal last delivery is retained.

2.6 Iterative local search and random variable neighborhood search heuristic

To test the VRP and SVRP formulations with varying cost functions, the formulations in Equations (2.4)-(2.12) were coded in Python and solved using Gurobi. Gurobi was set to solve the problems using a branch and bound approach [42] which can solve problems to guaranteed optimality but suffers from sharp increases in run-time with increase in the size of the solution space. Due to the addition of the superscript k in the solution space for x, optimally solving the SVRP on networks bigger than 20 nodes began leading to relatively long computational times. Therefore, a heuristic method for solving the SVRP with different objective functions is utilized. We develop a heuristic based on a random variable neighborhood search (RVNS) within an iterative local search (ILS) framework. The authors in [39] showed promising results when using a RVNS local search within a local iterative search framework when

applied to VRP instances. The authors in [1] redeveloped this structure for the SVRP, implemented some modified neighborhood structures, and changed the insertion procedure for reintroducing removed nodes to the solution. Both previous versions of ILS with RVNS focused solely on the minimum cost objective, but they both showed promise over other heuristics such as tabu search, genetic algorithms, and construction algorithms.

A pseudo-algorithm is provided, Algorithm 1, to show the overall flow of the heuristic. The main idea of this heuristic is to generate an initial solution and then use the RVNS procedure to locally improve this solution. Before the initial solution is generated, the optimal solution is set to infinity, see line 1 where $f(z_c, z_{ld}, z_{ad}) \triangleq \lambda_1 z_c + \lambda_2 z_{ld} + \lambda_3 z_{ad}$, and we call the optimal solution vectors returned by the algorithm as x^*, y^*. The main algorithm, lines 2-22, is performed a user-defined number of times, denoted $num\_iterations \in \mathbb{N}$, which signifies the iterative portion of the heuristic. The first step is to generate an initial solution, line 3, and this procedure is further discussed later. Then, line 4, the initial solution is set as the current local optimal solution, denoted $\tilde{x}, \tilde{y}$. The main local search part of the heuristic is performed in lines 6-16, wherein the RVNS is performed on the current solution. If this current solution is better than the best local solution, the best local solution is updated, lines 8-11. After the RVNS is exhausted, the solution is perturbed using the perturbation mechanism described in Algorithm 4, line 15. The process of performing the RVNS and perturbation is repeated until the solution has not improved $max\_not\_improved$ number of times. Finally, the current locally optimal solution is compared with the global optimal solution, and the best solution is saved as the global optimum.

The framework for the heuristic is the same as [1]. The heuristic in the cited paper is designed for minimum cost objective, $\lambda_1 = 1, \lambda_2 = \lambda_3 = 0$. However, this structure did not provide satisfactory results with the min-max objective function, or $\lambda_2 = 1$ and $\lambda_1 = \lambda_3 = 0$. The differences in the algorithms are in the RVNS, where the proposed algorithm differs in how nodes are selected for neighborhood operations. More details are provided in Section 2.6.2.

Algorithm 1: ILS + RVNS Framework.

Result: x^*, y^*.

```
1  Set f*(z_c, z_ld, z_ad) → ∞;
2  for i = 1, ··· , num_iterations do
3      [x, y] = Initial_Solution(N, d, m, cost_fun);
4      x̃ = x, ỹ = y;
5      not_improved = 0;
6      while not_improved < max_not_improved do
7          [x, y] = RVNS(x̃, ỹ);
8          if f(z_c, z_ld, z_ad) < f(z̃_c, z̃_ld, z̃_ad) then
9              x̃ = x, ỹ = y;
10             not_improved = 0;
11         end
12         else
13             not_improved += 1;
14         end
15         [x, y] = Perturbation(x̃, ỹ)
16     end
17     if f(z̃_c, z̃_ld, z̃_ad) < f*(z_c, z_ld, z_ad) then
18         x* = x̃;  y* = ỹ;
19         f*(z_c, z_ld, z_ad) = f(z̃_c, z̃_ld, z̃_ad)
20     end
21 end
```

2.6.1 Developing initial solutions

In this section, the method for generating the initial solutions in line 3 of Algorithm 1 is discussed. In this paper, we are assuming the number of vehicles is given by the user. The first step in generating the initial solutions is to assign the first node to each vehicle. Since multiple objective functions are considered, the selection of the initial nodes is allowed to depend on the cost function. For the minimum cost objective, the first nodes are inserted in order of increasing cost from distance to the depot. This is an intuitive decision since the best solutions to many routing instances involve vehicles doing short tours close to the depot. For the last delivery objective function, the vehicles are inserted in descending order starting from maximum distance from the depot. The optimal solution for minimum largest last delivery, if feasible, will be an out-and-back tour to the farthest node. After inserting

the first node to every route, the rest of the nodes are inserted in the same order as above, but the procedure for determining the cheapest cost differs. For this heuristic, two different insertion criteria are used, and the criteria is randomly selected at each iteration. The first is the nearest neighbor procedure. In this case, the nearest node, by minimum cost, that has been visited by a vehicle with remaining capacity to service the newest node to be inserted is chosen. The same vehicle is then used to visit the new location, and the new node is inserted into the route directly after the nearest neighbor. The second insertion criteria is given by the cheapest insertion of new node k between nodes i, j according to the following,

$$u(n_v) = (C_{ik}^{n_v} + C_{kj}^{n_v} - C_{ij}^{n_v}) - \gamma(C_{0k}^{n_v} + C_{k0}^{n_v}) \tag{2.29}$$

where $u(n_v)$ is the insertion cost, $\gamma > 0$ is a weighting cost on the back and forth trips to the depot, and n_v denotes the current vehicle being considered. This equation is considered for all vehicles, and the minimum $u(\cdot)$ value leads to insertion of the new customer k into the mth route between nodes i and j. It is noted that this procedure will always find a feasible solution to the SVRP as the available capacity is assumed to be greater than or equal to the total demands. In the case that the VRP initial solution is infeasible, the procedure is reinitialized, and if it fails more than 5 times, the procedure randomly assigns nodes until a feasible solution is generated.

Algorithm 2: Initial_Solution(N, d, m, cost_fun)

Result: $[x, y]$, Feasible Initial Solution

```
unvisited = sort(Nodes, cost_fun);
for All vehicles do
    [x, y] =Assign_first_node(unvisited, cost_fun);
    Remove_Assigned_node(unvisited);
end
while unvisited != Empty do
    procedure = Select_random_insertion_procedure(nearest, cheapest);
    [x, y] = Insert_node(procedure, x, y);
    Remove_Assigned_node(unvisited);
end
```

2.6.2 Random variable neighborhood search

Once the initial solutions are generated in Algorithm 2, they are modified by a random variable neighborhood search procedure. For the purpose of this work, we consider a neighborhood as a feasible solution to the mixed integer program that is "close" to the current solution. The solutions are considered neighbors since only a small difference such as two nodes being swapped within a route or a node being shifted to a different route exists between solutions. Neighborhood operations will be used to modify the current solution to these modified solutions, and the results will be checked to see if improvements in the objective function have been made.

Due to the size of the problem and solution space, specifically within the SVRP case, it becomes very impractical to consider every possible neighborhood operation at each step in the process. For this reason, a random variable neighborhood search is used which randomly selects neighboring solutions from the neighborhood structure. Many of the applications of RVNS use truly random selection of nodes and operations, but in this heuristic we will use a probabilistic selection process in some instances. There are a total of six neighborhood operations that are considered in this work, and four of them are identical for both the VRP and SVRP. These 4 common neighborhood operations are:

- Shift 1: One node, i, is shifted from route x^1 to route x^2.
- Shift 2: An arc of adjacent nodes, $i \to j$, is shifted from route x^1 to route x^2. The nodes can be inserted as either $i \to j$ or $j \to i$, and the solution that increases the cost for x^2 the least is selected.
- Swap 1: Node i from route x^1 is swapped with node j in route x^2. For the VRP, it is checked to make sure that capacity is not violated by this swap. If capacity is violated, the procedure is aborted. For the SVRP, depending on the demands at i, j, k, splits may be utilized if they reduce the overall cost, see [1].

- Cross: The arc $i \to j$ from route x^1 and arc $k \to l$ from route x^2 are removed. These two arcs are reinserted into the solution with cheapest cost while also considering the reverse arcs, $j \to i$ and $l \to k$, respectively.

In all of the above instances, it is assumed that the vehicles assuming new nodes within their routes can also successfully accommodate all of the demands as well.

For the VRP, an additional procedure is included, Remove Node, wherein a node i is removed from the solution and reinserted with lowest cost. The vehicle previously servicing customer i is considered as tabu, and if feasible is not allowed to re accept node i in its route. The SVRP case includes a similar procedure, called K-split [1], wherein a node is removed from all routes in the solution, and the node is reinserted using the insertion procedure described in Algorithm 3. For this insertion procedure, the cost of adding the node to any route that has residual capacity greater than 0 is considered. The insertions are sorted in order of increasing cost $u_{n_v} = \tilde{c}_i/q_{n_v}$ where $\tilde{c}_i$ is the cheapest cost of inserting node i into route n_v, and q_m is the residual capacity of route n_v before visiting customer i. This procedure is similar to the greedy knapsack heuristic which the authors in [43] showed is a very fast and accurate approximation. If the total residual capacity is less than the demand at node i, then the procedure is aborted. For the SVRP case, there will not be any feasibility issues, but in the case of the VRP, if there is no feasible reinsertion due to limited available capacity of vehicles, the procedure is restarted with a different node.

One of the key features of the algorithm must create is that it needs to work well for both the SVRP and the VRP for both minimum cost and minimum last delivery objective functions. The iterative local search plus RVNS has already shown good results for minimum cost objective for the VRP [39] and the SVRP [1]. For both of these cases, the RVNS is random, and the nodes and routes selected in the neighborhood operations are chosen randomly. In the last delivery case, it is intuitive to see that not every node and route contributes directly to the cost function, and for this reason the node and vehicle selections

Algorithm 3: Insertion(x, y, q, $Node_i$)

Result: Node i added to the solution and demand at i fulfilled.

```
for i = 1, ··· , Vehicles_with_available_capacity do
    c̃_i = Cheapest_Feasible_Insertion(x^i, Node_i);
    u_i = c̃_i / q_i;
end
sort(u);
while Demand at node_i is unmet do
    [x, y] =Insert node i into the route with smallest u value;
    Update(u);
end
```

have been biased. In the case of Shift 1, Shift 2, and Remove Node or K-split, a bias is introduced to select either vehicles or nodes that are part of routes with large last deliveries. As an example, for the Shift 1 operation, the last delivery times of each vehicle are calculated. These values are then normalized by the largest last delivery such that the largest value is 1. Next, all of the values are multiplied by some user-defined integer value, for example 1000 and rounded to the nearest integer, and this value for each vehicle determines the number of entries into the selection process. A random number between 1 and the number of entries is generated, and the route corresponding to that number is the vehicle chosen to have a node shifted from its solution.

Example 2.9. To illustrate how this setup can bias the selection of routes with larger last delivery, consider the following example where there are 5 routes with the following last delivery times, $ld^1 = 65, ld^2 = 110, ld^3 = 55, ld^4 = 114, ld^5 = 175$. Normalizing these routes and scaling by 1000, and then determining percentages gives the following percentage of being selected: It can be seen that in this case, route 5 which had the largest last delivery time, and is actually the value of the objective function, is now more than 3x as likely to be selected within the RVNS as route $n_{\text{veh}} = 3$, which is the shortest. △

The RVNS is a local heuristic which has a downside of getting stuck in local optimum. To overcome this limitation, after exhausting the RVNS, the current best solution is perturbed

Table 2.9: Example of biased route selection under last delivery objective.

Vehicle Number	Percentage of Selection
1	12.5%
2	21.2%
3	10.6 %
4	22.0 %
5	33.7 %

by removing a random set of k nodes from the solution. This procedure is shown in Algorithm 4, and the key component to the mechanism is that the nodes to be reinserted are sorted in descending order by their distances from the depot. The nodes farther from the depot are inserted first as they generally have greater effect on the objective function. These nodes are then reinserted into the solution using the same insertion procedure in Algorithm 3. The authors in [1] showed the best results occurred when a random number of nodes between five and seven were used in the perturbation, so we choose the same values.

Algorithm 4: Perturbation(x, y, k, q)

Result: Perturbed solution x, y

```
Node_list = Select_k_random_nodes_to_remove(k);
[x, y] = Remove_k_nodes_from_solution(x, y, q,Node_list);
sort_k_nodes(Node_list);
for i = 1 : k do
    [x, y] = Insertion(x, y, q, Node_list_i);
end
```

2.7 Results & analysis on benchmark datasets

The goal of this section is to numerically examine some of the relationships proposed in Sections 2.3, 2.4, and 2.5 regarding the VRP and SVRP for different objective functions. First, we validate the use of the proposed heuristic by showing that solutions within 1% of best-known solutions can be found for both the SVRP and VRP formations with minimum cost objective. A few examples are provided where the optimal solutions to SVRP and VRP

with last delivery objective are discussed since benchmark sets with utilizing this objective function are very limited. Next, 3 benchmark instances are utilized to compare the solutions of the proposed formulations with both objective functions. It is known that the ratio of average demand to vehicle capacity can play a significant role in the validity of split deliveries improving the minimum cost objective function, so more simulations are provided where the vehicle capacities used in the benchmark instances are changed.

2.7.1 Comparison of developed heuristic with some best known results

Table 2.10: The result of comparing the proposed SVRP heuristic solutions for minimum cost objective versus the results in [1] for the Belenguer dataset.

Belenguer	$c(SplitILS_c)$	$c(SVRP_c)$	% Difference
eil22	375.28*	375.28*	0
eil23	568.56*	568.56*	0
eil30	512.72*	512.72*	0
eil33	837.06	837.07	0.001
eil51	524.61	524.61	0
eilA76	823.89	827.65	0.46
eilC76	739.83	746.73	0.93
eilD76	688.37	693.71	0.78
eilA101	826.26	834.13	0.95
eilB101	1078.58	1100.96	2.07
S51D1	459.5	461.21	0.37
S51D3	949.96	959.70	1.02
S76D1	598.98	603.06	0.68
S76D3	1429.01	1451.33	1.56
S101D1	728.44	739.27	1.49
		Average	0.69

In order to first verify that the proposed heuristic provides results that are comparable to the state of the art, we test the SVRP on two benchmark datasets, namely the Belengeur dataset from [44] and the Archetti dataset from [36]. Both of these instances were tested in [1] where the basis of this algorithm is generated from. The Belenguer data set contains instances between 22 and 101 nodes with varying demands where the node locations were

generated from instances in the TSPLIB [45] . The instances beginning with "eil" are the TSPLIB instances with the stated number of nodes including the depot. The instances beginning with "S" denote adapted instances to vary the demands of the TSPLIB cases. The demands are randomly generated in the intervals as follows D1 = $[0.01q, 0.1q]$, D2 = $[0.1q, 0.3q]$, D3 = $[0.1q, 0.5q]$, D4 = $[0.1q, 0.9q]$, D5 = $[0.3q, 0.7q]$, and D6 = $[0.7q, 0.9q]$. In the chosen Archetti dataset, we choose 3 main instances, p01 with 50 nodes, p02 with 75 nodes, and p03 with 100 nodes. The demands include the same notation as the Belenguer with the exception of the cases which are not labeled for demands. In these cases, the demands are random. While fast solve times are fundamental characteristics for heuristics, the goal of this chapter is not to develop the fastest and most accurate heuristic, but we need to develop a heuristic that gives good solutions to all of our formulations and objective functions in reasonable time. To ensure reasonable solve times, a maximum of 120s is allowed for each instance. For this reason, it will be seen that for smaller instances of less than 50 or so nodes, this algorithm performs very well. For larger instances of 100 nodes or more, the time limit is reached in many cases, but we show that solutions within 1-2% of state of the art methods can be achieved.

The parameters used for the simulation are $num\_iterations = 10$, $max\_not\_improved = 1500$, number of nodes in perturbation is $5-7$, 3 simulations are run, and the average solution is computed. Note that for SplitILS, $num\_iterations = 10$, $max\_not\_improved = 6(m \cdot n)$, where n denotes the number of nodes, m denotes the number of routes, the number of nodes in perturbation is a random integer between 5 and 7, and 3 simulations are run.

As seen in Table 2.10, the proposed heuristic generates solutions on average within 0.69% of the average solution generated by the SplitILS. Table 2.11 shows the proposed heuristic and the results of SplitILS on the Archetti dataset for instances up to 100 nodes. In this case, the average solution is 1.20% worse than the solutions of SplitILS. These results show that this heuristic is competitive with one of the best known heuristics for the SVRP with minimum cost objective. However, this heuristic also must seamlessly solve VRP instances

Table 2.11: The result of comparing the proposed heuristic SVRP solutions for minimum cost objective versus the results in [1] for the Archetti dataset.

Archetti	$c(SplitILS_c)$	$c(SVRP_c)$	% Difference
p01	524.60*	524.60*	0
p01_D2	760.7*	765.95	0.69
p01_D3	1005.93	1016.83	1.08
p01_D4	1489.05	1506.24	1.15
p01_D5	1484.62	1501.50	1.14
p01_D6	2160.6	2177.27	0.77
p02	824.39	837.56	1.60
p02_D2	1112.7	1123.73	0.99
p02_D3	1503.42	1540.31	2.45
p02_D4	2304.89	2325.06	0.88
p02_D5	2222.58	2254.13	1.42
p02_D6	3226.79	3266.72	1.24
p03	826.45	835.03	1.04
p03_D2	1462.37	1484.79	1.53
p03_D3	2001.83	2041.33	1.97
p03_D4	3155.22	3179	0.75
p03_D5	2991.89	3052.65	2.03
p03_D6	4389.17	4423.66	0.79
		Average	1.20

as well, and 2 benchmark instances are utilized to verify the effectiveness of the proposed heuristic.

To verify the effectiveness of the heuristic for VRP instances, two benchmark datasets are used, namely Augerat sets A and B [46]. Both of these datasets contain instances with somewhere between 31 and 78 nodes including the depot. These datasets include best known solutions, many of which are optimal solutions we denote with a $(\cdot)^*$. The main difference between the two sets of instances is that the nodes in A are randomly distributed, while the nodes in the B dataset are clustered. The instance name includes the number of nodes followed by k and a number to denote the minimum number of vehicles required to fill the demands. Table 2.12 shows the result of running the proposed heuristic on Augerat A dataset. The average solution is 0.39% worse than the best known solution, and the proposed algorithm found 5 optimal solutions. Table 2.13 shows the result of running the proposed

Table 2.12: The result of comparing the proposed VRP heuristic solutions for minimum cost objective versus best known results for Augerat A dataset.

Augerat A	$c(SVRP_c)$	$c(Best)$	% Difference
An32k5	784*	784*	0
An33k5	661*	661*	0
An33k6	743	742*	0.13
An34k5	778*	778*	0
An36k5	799*	799*	0
An37k5	669*	669*	0
An37k6	950	949*	0.11
An38k5	730*	730*	0
An39k5	825	822*	0.36
An39k6	833	831*	0.24
An44k7	944	937*	0.75
An45k6	976	944*	3.39
An45k7	1152	1146	0.52
An46k7	919	914*	0.55
An48k7	1073*	1073	0
An53k7	1021	1010*	1.09
An54k7	1167	1167	0
An55k9	1074	1073*	0.09
An60k9	1379	1408	-2.06
An61k9	1042	1035	0.68
An62k8	1304	1290	1.09
An63k9	1650	1634	0.98
An63k10	1328	1315	0.99
		Average	0.39

heuristic on Augerat B dataset. The average solution is 0.43% worse than the best known solution, and the proposed algorithm also found 5 optimal solutions in this case.

Validating the proposed heuristic for use with the last delivery objective proved more difficult as most of the benchmark datasets are only tested for minimum cost objective. All of the instances with less than 40 nodes in the Augerat A and B datasets were solved using Gurobi for both the SVRP and VRP cases for minimum last delivery objective. In all cases, the optimal last delivery is achieved by the proposed heuristic. While it is unlikely that the optimal will be found for larger instances for all cases, we will assume that the solutions achieved by the proposed heuristic are good enough for our comparisons.

Table 2.13: The result of comparing the proposed VRP heuristic solutions for minimum cost objective versus best known results for Augerat B dataset.

Augerat B	$c(SVRP_c)$	$c(Best)$	% Difference
Bn31k5	672*	672*	0
Bn34k5	788*	788*	0
Bn35k5	955*	955*	0
Bn38k6	806	805*	0.12
Bn39k5	549*	549*	0
Bn41k6	829*	829*	0
Bn43k6	744	742*	0.26
Bn44k7	909*	909*	0
Bn45k5	764	751*	1.73
Bn45k6	696	678*	2.65
Bn50k7	741*	741*	0
Bn50k8	1321	1313*	0.61
Bn51k7	1039	1032	0.68
Bn52k7	749	747*	0.27
Bn56k7	710	707*	0.42
Bn57k9	1608	1598	0.625
Bn63k10	1541	1537	0.26
Bn66k9	1333	1374	-2.98
Bn67k10	1073	1033	3.87
Bn68k9	1296	1304	-0.61
Bn78k10	1267	1266	0.08
		Average	0.43

2.7.2 Results on benchmark instances

The first simulations performed involved using the Belenguer, Augerat A, and Augerat B instances as is. In these cases, the minimum number of vehicles to fill the demand are used, which we will call tight capacity. In all cases, the SVRP and VRP approach are utilized for both objective functions, and the data is provided in Tables 2.14, 2.15, and 2.16.

The first question to consider is whether or not using split deliveries tends to improve the solutions. As can been in the first column of Tables 2.14, 2.15, and 2.16, when using the cost function objective for both classical and split VRP, the SVRP averages a 3% improvement in Belenguer instances, a 1% improvement for Augerat A instances, and a 2% improvement for the Augerat B instances. The SVRP does provide improvement on average for all cases

Table 2.14: Dataset of comparisons on Belenguer instances with tight capacity.

Belenguer	$\frac{c(VRP_c)}{c(SVRP_c)}$	$\frac{ld(VRP_c)}{ld(VRP_{ld})}$	$\frac{ld(VRP_{ld})}{ld(SVRP_{ld})}$	$\frac{c(VRP_{ld})}{c(VRP_c)}$	$\frac{c(VRP_{ld})}{c(SVRP_{ld})}$	$\frac{ld(SVRP_c)}{ld(SVRP_{ld})}$	$\frac{c(SVRP_{ld})}{c(SVRP_c)}$
eil22	1.01	1.09	1.00	1.09	1.04	1.09	1.06
eil23	1.00	1.49	1.14	1.25	1.12	1.70	1.12
eil30	1.05	1.59	1.12	1.08	1.04	1.46	1.09
eil33	1.00	1.67	1.00	1.14	0.98	1.38	1.17
eil51	1.05	1.50	1.00	1.18	1.20	1.03	1.03
eilA76	1.08	1.28	1.20	1.24	1.11	1.48	1.20
eilC76	1.01	1.35	1.01	1.18	1.00	1.24	1.19
eilD76	1.00	1.36	1.01	1.16	0.98	1.52	1.18
eilA101	1.02	1.30	1.02	1.22	1.00	1.49	1.23
eilB101	1.02	1.41	1.01	1.22	1.01	1.57	1.23
S51D1	1.00	1.30	0.98	1.16	1.02	1.18	1.14
S51D3	1.04	1.28	1.13	1.13	1.02	1.46	1.15
S76D1	1.02	1.41	1.06	1.20	1.08	1.24	1.13
S76D3	1.09	1.21	1.26	1.19	1.09	1.79	1.19
S101D1	1.09	1.23	1.04	1.21	1.05	1.05	1.26
Average	1.03	1.36	1.07	1.18	1.05	1.38	1.16
Min	1.00	1.09	0.98	1.08	0.98	1.03	1.03
Max	1.09	1.67	1.26	1.25	1.20	1.79	1.26

and a worse solution in only 1 of the instances, namely An48k7. The ratio of minimum last delivery objectives, proposed to have an upper bound of $2m - 1$ in Proposition 2.5, are found as 1.07 for the Belenguer instances, and 1.02 for both the Augerat A and B instances. As can be seen in the tables, there are more cases of greater than 5% improvement for last delivery times than for improvements in minimum cost objective. The data suggests that the cost of using the last delivery objective is actually more expensive for the VRP than the SVRP. While the SVRP finds better last delivery solutions, the cost is actually 5%, 0%, and 2% higher for the VRP for the Belenguer, Augerat A, and Augerat B instances, respectively.

It is also important to consider the effects of the different objective functions on the solutions under the same formulation. In many disaster response scenarios, the longest wait time of a customer can become paramount to their safety, and this may become the most important consideration, but we would like to know what size of cost this can incur. It can be seen in the 5th column of Tables 2.14, 2.15, and 2.16 that choosing the last delivery objective function increases the total cost by an average of 18%, 20%, and 19% for the

Table 2.15: Dataset of comparisons on Augerat A instances with tight capacity.

Augerat A	$\frac{c(VRP_c)}{c(SVRP_c)}$	$\frac{ld(VRP_c)}{ld(VRP_{ld})}$	$\frac{ld(VRP_{ld})}{ld(SVRP_{ld})}$	$\frac{c(VRP_{ld})}{c(VRP_c)}$	$\frac{c(VRP_{ld})}{c(SVRP_{ld})}$	$\frac{ld(SVRP_c)}{ld(SVRP_{ld})}$	$\frac{c(SVRP_{ld})}{c(SVRP_c)}$
An32k5	1.00	1.83	1.00	1.25	0.99	1.65	1.27
An33k5	1.01	1.65	1.00	1.21	1.02	1.54	1.19
An33k6	1.00	1.73	1.01	1.16	1.00	1.61	1.16
An34k5	1.01	1.35	1.00	1.09	0.99	1.37	1.11
An36k5	1.01	1.42	1.03	1.24	1.03	1.60	1.21
An37k5	1.00	1.75	1.01	1.20	0.97	1.79	1.24
An37k6	1.00	1.90	1.02	1.22	0.99	1.78	1.23
An38k5	1.01	1.48	1.04	1.17	0.99	1.41	1.19
An39k5	1.01	1.56	1.00	1.17	0.98	1.55	1.20
An39k6	1.01	1.62	1.03	1.24	1.01	1.82	1.24
An44k7	1.01	1.72	1.10	1.16	0.96	1.70	1.22
An45k6	1.05	1.34	1.06	1.18	1.01	1.58	1.22
An45k7	1.01	1.76	0.99	1.24	1.01	1.67	1.23
An46k7	1.01	1.58	0.99	1.23	1.05	1.76	1.18
An48k7	0.99	1.46	1.01	1.22	1.01	1.50	1.20
An53k7	1.03	1.50	1.03	1.24	1.01	1.39	1.21
An54k7	1.00	1.39	1.05	1.20	1.05	1.62	1.11
An55k9	1.00	1.38	0.99	1.16	0.94	1.37	1.27
An60k9	1.00	1.46	1.00	1.20	0.99	1.62	1.23
An61k9	1.02	1.33	1.13	1.31	1.09	1.48	1.22
An62k8	1.00	1.76	1.00	1.19	0.98	1.91	1.21
An63k9	1.01	1.54	1.02	1.22	1.00	1.53	1.23
An63k10	1.01	1.65	1.01	1.20	0.97	1.66	1.25
Average	1.01	1.57	1.02	1.20	1.00	1.60	1.21
Min	0.99	1.33	0.99	1.09	0.94	1.37	1.11
Max	1.05	1.90	1.13	1.31	1.09	1.91	1.27

Belenguer, Augerat A, and Augerat B instances, respectively. Similarly, choosing the cost function as an objective increases the last delivery time by an average This does, however, provide an decrease of last delivery time by approximately 36%, 57%, and 61% respectively for the instance cases. We see similar results for SVRP, and choosing the last delivery objective function increases the average total cost by 16%, 21%, and 19% respectively. By choosing the SVRP and the minimum cost objective over the last delivery objective, the last delivery increases by averages of 38%, 60%, and 52% respectively.

A key observation made during this first set of simulations is that the ratio of average demand to vehicle capacity has a significant influence on whether or not using split deliveries

Table 2.16: Dataset of comparisons on Augerat B instances with tight capacity.

Augerat B	$\frac{c(VRP_c)}{c(SVRP_c)}$	$\frac{ld(VRP_c)}{ld(VRP_{ld})}$	$\frac{ld(VRP_{ld})}{ld(SVRP_{ld})}$	$\frac{c(VRP_{ld})}{c(VRP_c)}$	$\frac{c(VRP_{ld})}{c(SVRP_{ld})}$	$\frac{ld(SVRP_c)}{ld(SVRP_{ld})}$	$\frac{c(SVRP_{ld})}{c(SVRP_c)}$
Bn31k5	1.00	1.28	1.00	1.14	1.00	1.28	1.14
Bn34k5	1.01	1.43	1.03	1.13	1.07	1.47	1.07
Bn35k5	1.00	1.13	1.00	1.16	1.01	1.13	1.15
Bn38k6	1.00	1.47	1.00	1.17	1.06	1.47	1.11
Bn39k5	1.00	1.29	1.00	1.31	0.95	1.43	1.37
Bn41k6	1.00	2.01	1.01	1.07	1.00	2.03	1.07
Bn43k6	1.00	1.99	1.02	1.16	1.00	2.03	1.16
Bn44k7	1.00	2.22	1.02	1.17	1.03	2.27	1.13
Bn45k5	1.02	1.77	1.01	1.18	1.00	1.25	1.20
Bn45k6	1.03	1.41	1.00	1.18	0.97	1.48	1.25
Bn50k7	1.00	1.12	1.00	1.09	1.02	1.12	1.07
Bn50k8	1.01	1.73	1.00	1.23	1.00	1.68	1.24
Bn51k7	1.02	1.31	1.10	1.22	1.09	1.38	1.14
Bn52k7	1.00	1.72	1.02	1.18	0.99	1.36	1.18
Bn56k7	1.00	1.42	1.00	1.36	0.99	1.42	1.38
Bn57k7	1.19	2.09	1.11	1.01	1.05	1.22	1.15
Bn57k9	1.00	1.45	1.00	1.15	1.02	1.51	1.12
Bn63k10	1.03	1.81	1.03	1.23	1.13	1.44	1.13
Bn64k9	1.09	1.56	1.00	1.16	1.08	1.67	1.18
Bn66k9	1.00	1.80	1.09	1.24	1.02	1.55	1.22
Bn68k9	1.02	1.57	1.05	1.16	0.95	1.57	1.25
Bn78k10	1.01	1.32	1.02	1.28	1.02	1.70	1.27
Average	1.02	1.61	1.02	1.19	1.02	1.52	1.19
Min	1.00	1.12	1.00	1.01	0.95	1.12	1.07
Max	1.19	2.22	1.11	1.36	1.13	2.27	1.38

tends to improve the solution. The Augerat A and B instances have an average demand of just over 13% of vehicle capacity. On the contrary, cases such as S51D3 and S76D3 in the Belengeur instances, which have average demands equal to 28% and 29% of vehicle capacity, respectively. In both of these cases, it can be seen that the SVRP performs better in both cost objective, 4% and 9%, and last delivery objective, 13% and 26%. This observation most likely contributes to the 7% improvement on last delivery objective for the SVRP on Belenguer instances.

After observing the cases with higher demand ratios in the Belengeur cases leading to larger variations in the ratios of cost evaluations, an extra set of simulations is run on the Archetti instances. To make the instances feasible for the VRP, enough vehicles are added

to ensure that the problems become feasible. The results are shown in Table 2.17. The first major takeaway is that the benefits of using the SVRP framework over the VRP is immediately seen in the first column. For minimum cost objective, the SVRP is better by an average of 30% and with a maximum improvement of 72%. The last delivery times, however, are almost identical for the two formulations when both optimize for minimum last delivery. This may be occurring since there are a large amount of vehicles, and in all of the cases of D2,D3,D4,D5, and D6, the minimum last delivery is found as the maximum distance from the depot to a node, which is optimal. As this occurs for both VRP and SVRP, there is no possibility for the SVRP to improve over the VRP. For the VRP, the cost when using the last delivery objective is on average 19% more expensive than the minimum cost objective. The last delivery for the VRP increases by 56% when using the minimum cost objective instead of the minimum last delivery objective. It can be seen that choosing the correct objective functions becomes even more important for the SVRP in these high demand scenarios. In this formulation, the cost of using the last delivery objective is 53% higher than when minimizing cost. Furthermore, the last delivery is 89% higher on average when minimizing cost instead of minimizing last delivery, and there are 4 instances where it is over 200%. This suggests that the user should carefully consider the scenario when choosing the correct objective function. If the user chooses to minimize total cost, we have seen that the last customer may have to wait twice as long as the case in which last delivery objective is considered. In cases of emergencies, this type of gap could be the difference between whether or not someone is saved.

An additional dataset from [47] is used to test the more extreme case that demands are always greater than 50% vehicle capacity. The nodes in this dataset are distributed in concentric circles around the depot. Instead of increasing vehicle capacity as in the previous two cases, we increase the number of vehicle to allow feasibility for the VRP. Because the demands are all above half vehicle capacity, the number of vehicles is equal to the number of nodes, and optimal solutions can be readily computed as the sum of out-and-back tours to

Table 2.17: Dataset of comparisons on Archetti instances with 50% higher vehicle capacity.

Archetti	$\frac{c(VRP_c)}{c(SVRP_c)}$	$\frac{ld(VRP_c)}{ld(VRP_{ld})}$	$\frac{ld(VRP_{ld})}{ld(SVRP_{ld})}$	$\frac{c(VRP_{ld})}{c(VRP_c)}$	$\frac{c(VRP_{ld})}{c(SVRP_{ld})}$	$\frac{ld(SVRP_c)}{ld(SVRP_{ld})}$	$\frac{c(SVRP_{ld})}{c(SVRP_c)}$
p01_00	1.15	1.38	0.99	1.09	1.02	1.92	1.23
p01_D2	1.39	1.50	0.96	1.10	0.99	1.92	1.55
p01_D3	1.61	1.57	1.00	1.01	0.95	1.84	1.73
p01_D4	1.20	2.05	1.00	1.35	1.04	1.80	1.56
p01_D5	1.22	1.50	1.00	1.26	0.94	1.80	1.63
p01_D6	1.23	1.09	1.00	1.24	1.00	1.68	1.52
p02_00	1.33	1.67	0.97	1.06	0.98	2.37	1.44
p02_D2	1.55	1.87	1.00	1.05	1.00	2.20	1.64
p02_D3	1.13	1.93	1.00	1.41	0.96	2.28	1.66
p02_D4	1.19	1.81	1.00	1.34	1.02	1.74	1.55
p02_D5	1.16	1.65	1.00	1.34	1.03	2.02	1.52
p02_D6	1.22	1.23	1.00	1.28	0.99	1.67	1.58
p03_00	1.14	1.52	1.00	1.17	0.98	1.78	1.36
p03_D2	1.59	1.50	1.00	1.10	1.03	1.96	1.71
p03_D3	1.72	1.90	1.00	0.93	1.08	1.88	1.48
p03_D4	1.18	1.36	1.00	1.23	0.95	1.68	1.52
p03_D5	1.18	1.44	1.00	1.21	0.99	1.78	1.45
p03_D6	1.21	1.20	1.00	1.20	1.00	1.62	1.45
Average	1.30	1.56	1.00	1.19	1.00	1.89	1.53
Min	1.13	1.09	0.96	0.93	0.94	1.62	1.23
Max	1.72	2.05	1.00	1.41	1.08	2.37	1.73

each node. To additionally allow the instances to favor the SVRP, an additional node with demand of 1 is added at 1 unit from the depot to absorb extra vehicles. The results can be seen in Table 2.18, and the VRP on average costs 17% more than the SVRP, and more importantly this is for optimal VRP solutions. This data suggests that using split deliveries is even more useful in cases of higher demand which was also showed by the authors in [48]. Exploring the last delivery objective is less interesting in these cases as the VRP formulation has the optimal solution by default as there is only one solution to the problem, which has out and back tours to all nodes.

Overall, it seems that using the SVRP formulation can lead to lower costs when minimizing cost, but this is especially true for networks that have high average demand relative to vehicle capacity. We also showed that how different the last delivery time can be depending on the objective function, and in the case of routing where the customer waiting time is

Table 2.18: Comparing the SVRP to the VRP on the Chen dataset for high demand case.

Chen	$c(SVRP_c)$	$c(VRP_c)$	$\frac{c(VRP_c)}{c(SVRP_c)}$
SD2	716	800	1.12
SD3	439	480	1.09
SD4	648	720	1.11
SD5	1407	1600	1.14
SD6	849	960	1.13
SD7	3660	4400	1.20
SD8	5116	6240	1.22
SD9	2100	2400	1.14
SD10	2800	3220	1.15
SD11	13342	16800	1.26
SD12	7318	8800	1.20
SD13	10254	12480	1.22
SD14	10893	13200	1.21
SD15	15371	18720	1.22
		Average	1.17

considered, minimizing the last delivery time may be necessary. In the future, more studies can be done to extend this work to more comprehensive routing formulations including asymmetric cost functions, demands that are larger than vehicle capacity, and relaxing the constraint that the same number of vehicles must be used between the two different formulations. The fact that all vehicles must be used can be a detriment to the SVRP, and we postulate that relaxing this assumption would allow the SVRP to further outperform the VRP formulation. In the next chapter, some of the restrictions discussed in Remark 2.1 are relaxed to make the vehicle routing problem more suitable to UAVs.

Chapter 3

Applying vehicle routing approach to UAVs in the presence of wind

In some cases of practical interest, some assumptions on the vehicle routing problem, such as those employed in Chapter 2, may be insufficient to solve problems involving UAVs, specifically the multirotor UAVs considered in this work. Firstly, the range of the vehicles was considered as unlimited, and it is well-known that one of the current limitations of UAVs is their relatively short flight times, typically 30 minutes or less. Due to this limited flight time, it becomes increasingly necessary to include the capability of the vehicles to refuel or change batteries. UAVs also have a limited payload capacity compared to larger vehicles such as trucks, and assumptions that only one vehicle can visit each location may be unrealistic. Secondly, since the UAV must provide the thrust to carry this payload, the range of the UAV must not only be considered, but it may be a function of an optimization variable, namely the payload's mass. Thirdly, since the UAVs must ascend and descend at each customer location, there is a non-negligible service time during which the vehicle is consuming its valuable flight time. Finally, it is also common to exclude environmental factors such as wind, which can greatly affect UAVs, and even make seemingly feasible routes infeasible.

Researchers have been studying the problem of routing vehicles that have limited range with the need to refuel or recharge. The effects of vehicle size, capacity, and their ability to deliver payloads in an urban setting are studied in [49]. The Green vehicle routing problem [50] was developed to study the effects of allowing a fleet of heterogeneous vehicles to refuel

throughout their routes, effectively extending their range. However, the authors did not consider split deliveries or the capacities of the vehicles. The authors in [51] studied the problem where the fleet may be non-homogeneous and there may be time windows in which the customers must be visited. However, they do not consider split deliveries. In [52], the authors applied the split vehicle routing problem framework to the UAS routing problem where they considered the range of the vehicle as a linear function of the payload. In this formulation, the vehicles were allowed to recharge, but only at the depot and not at other sites.

Few authors have considered environmental factors such as wind when routing UAS. The authors in [53] account for wind by adjusting the cost matrix a priori and making linear assumptions on the vehicle dynamics. However, they only solve a TSP. In [54], it is shown that the optimal speeds of the UAS change when considering wind, and the results show that the vehicles may take drastically different routes to minimize energy consumption. However, neither of the cited methods include payload dependencies or refueling in their optimization problem.

In this chapter, a strategy is presented to construct a vehicle routing formulation whose cost matrix is built from the solution of a path planning problem, that is more conducive to UAV routing. The optimization approach is divided into two key steps, creating the cost matrix and formulating the optimization problem. In Section 3.1.1, the cost matrix of the vehicle routing problem will be constructed by utilizing an $A*$ path planning approach with a custom heuristic for guaranteeing some user-defined safety margins from obstacles. In Section 3.1.2, the generated paths are then adjusted for a known, constant wind velocity which is converted to a time-based objective function. In Section 3.1.3, a simplified model of power consumption for multirotor UAVs is presented, which is used to develop a linear energy versus payload constraint in the vehicle routing formulation. In Section 3.2, we present a vehicle routing problem designed for UAVs and whose cost matrix underlies the planned paths from the previous sections. A numerical simulation is provided in Section 3.3

to exemplify the results of both this work and the results of Chapter 2 on the importance of the objective function.

3.1 Cost matrix for UAV route optimization problem

The proposed optimization approach is developed in two main steps. The first step is the generation of a cost matrix, which determines the time it takes to safely traverse between different nodes. Due to the effects of wind on the velocity of UAVs, a constant wind can be assumed and accounted for in the cost matrix. A simplified power model of rotors is presented to determine the relationship between the power and the payload of the vehicle. The generated cost matrix and power versus payload relationship are then used in a vehicle routing approach for UAVs.

3.1.1 Using $A*$ path plans to generate possible routes

The cost matrix for many routing-type problems is given by the Euclidean distance between each pair of points within the graph. This leads to a convenient property, namely symmetry of the cost matrix, $C = C^{\mathrm{T}}$, where the cost is the same for going from node i to node j as it is going from node j to node i. Symmetry helps reducing computational times to find solutions, specifically in minimum cost solutions traversing the route forwards or backwards incurs the same cost when the matrix is symmetric. This approach may be effective when considering vehicles such as cars or trucks. For UAVs, however, there are additional factors, such as wind, that can make Euclidean distances a very poor estimate of time cost. In the near future UAVs may be delivering packages in large cities near buildings and obstacle avoidance with an additional safety margin should be included for the safety of both the vehicles and persons nearby. For this reason, the cost matrix in this work will contain feasible paths between all of the nodes that attempt to minimize the distance traveled

while still maintaining safe distance from obstacles.

In this chapter, a path planning approach based on the A^* framework has been chosen for its ability to generate paths that are optimal with respect to a user-defined cost function. The path planning problem for autonomous UAVs has been extensively studied in the literature including the use of $A*$ [55, 56], and its variants $D*$ [57] and $iADA^*$ [58]. The probabilistic roadmap method has the benefit of being a fast algorithm and guarantees low processing times by ignoring points that cannot be reached [59, 60]. This method, however, is not an optimization-based path planner, and runs anti-thesis to the goal of minimizing cost. Rapidly exploring random trees can also be implemented as fast obstacle avoidance path planners for UAVs [61, 62], but they do solve for optimal paths with respect to a user-defined cost function and do not apply to this work. Voroni diagrams can be used for UAV path planning [63], but inherently find the maximum distance from obstacles, which may produce paths which are very far from minimizing distance. While a mixed integer linear program, such as the one used for the main routing problem, could be used, the constraints are typically required to be convex, which can be a restrictive in many realistic environments. $A*$ does not require the free space or the obstacles to be convex, and hence can excel in planning in complex environments. A variant of this $A*$ heuristic is developed in [64] for tactical path planning in the presence of obstacles.

Next, the formulation and pseudo-algorithm for $A*$ are presented. Consider the map, $\mathcal{M}$, with the set of nodes $n \in V$ and arcs $(i,j) \in E$ for $i \in V, j \in V, i \neq j$, which are the same nodes and arcs used in the optimization problem. The initial node is denoted n_0, and the goal node is denoted n_{goal}. For a given node n, all nodes adjacent, or connected to n, are captured by V'. In our case, the graph is fully connected, and all nodes can be considered as adjacent, so it is a tunable parameter for us to determine how many nodes are considered as neighbors to any 1 node. For now, consider the neighbors of a given node n as the set of neighbors which are 1 node distance away. Denote the set of open nodes as O, which contain the quadruplets $(n, g(n_0, n), f(n)$, Pointer), and O is sorted according the values of

$f(\cdot)$. Open nodes are nodes that are adjacent to nodes that have already been visited by the algorithm. Closed nodes contain both the nodes that have already been visited by the algorithm and those that have yet to be opened. The function $f(n)$ captures the sum of the cost to go from the start, n_0, to node n, denoted $g(n_o, n)$, plus a heuristic estimate of the remaining cost to get to the goal. The pointer is a variable which denotes the node that was used to open the current node in previous iterations. This is useful for tracing the path from the goal to the start. Since planning is assumed to be around obstacles, let the occupied nodes in the graph be captured by the set $\tilde{O}$.

Algorithm 5: A^* algorithm for returning shortest path w.r.t. $g(\cdot, \cdot)$.

Result: Set of nodes V^* which denote the path of lowest cost from n_0 to n_{goal}.

```
Empty(O);
Load map M, heuristic parameters λ1, λ2, λ3, and goal n_goal ;
Insert (n_0, g(n_0, n_0), f(n_0), Pointer) → O ;
while !Empty(O) do
    Set current Node n to node n with Minimum f(n) in O ;
    if n = n_goal then
        Break while and Reconstruct Path;
    end
    Remove {n, g(n_0, n), f(n), Pointer} from O ;
    for V' adjacent to n do
        ñ =current(V');
        if ñ ∉ O then
            Insert (ñ, g(n_0, ñ), f(ñ), Pointer) → O ;
            In O, set pointer of ñ towards n ;
        end
        else if g(·, ñ) > g(n_0, n) + g(n, ñ) then
            Modify O by setting pointer of ñ towards n ;
            Update (ñ, g(n_0, ñ), f(ñ), Pointer) → O;
        end
    end
end
Reconstruct Path(O) – Trace the pointers from the goal back to the start;
```

The *cost function* for the proposed A^*-based path planning is given by

$$f_k \triangleq g_k + h_k, \qquad k \in \overline{\mathbb{N}}, \tag{3.1}$$

where

$$g_k \triangleq \sum_{q=1}^{k} \left[\kappa(\mathrm{d}_2(\hat{r}_q, \tilde{O}))\mathrm{d}_2(\hat{r}_q, \hat{r}_{q-1}) \right], \tag{3.2}$$

denotes the *cost-to-come function*,

$$h_k \triangleq \mathrm{d}_2(\hat{r}_k, n_{\mathrm{goal}}), \tag{3.3}$$

denotes the *heuristic function*,

$$\kappa(\alpha) \triangleq 1 + \frac{\mu_1}{\mu_2 + e^{\mu_3 \alpha}}, \qquad \alpha > 0, \tag{3.4}$$

denotes the *weighing function*, and $\mu_1, \mu_2, \mu_3 > 0$ are user-defined parameters. This function

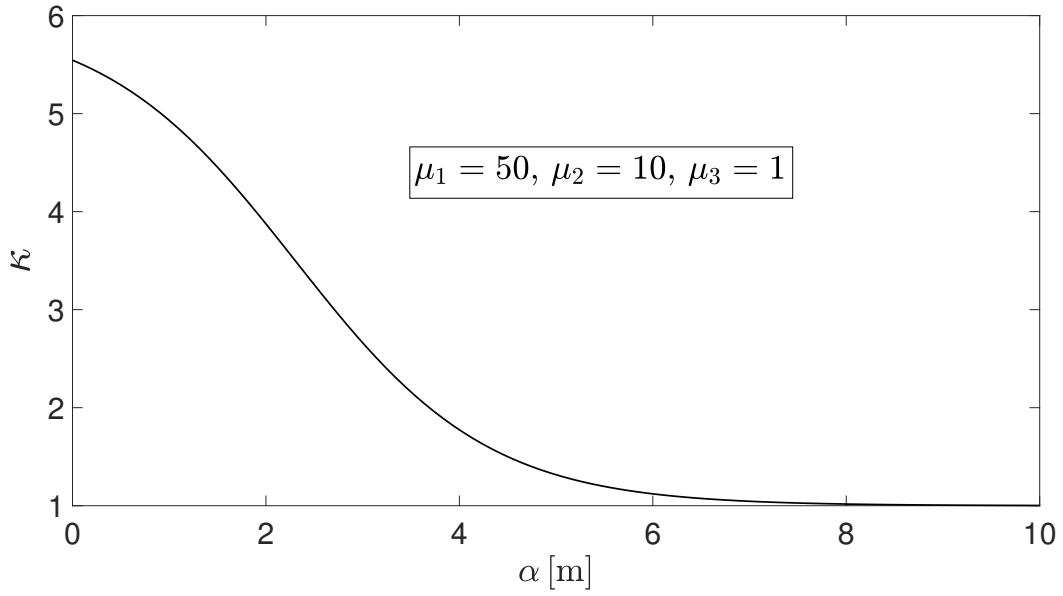

Figure 3.1: Plot of $\kappa(\alpha)$ function used in $A*$ heuristic.

has a maximum at 0, and hence strongly discourages the path planner from producing points close to the obstacles. The function is strictly decreasing and tends to 1 as $\alpha \to \infty$ which suggests that nodes that are far away from obstacles have little influence other than the

distance function, and we get a shortest path behavior in this case. The distance function is also slightly modified to discourage changes in altitude as we consider it safer for the UAVs to fly between waypoints at near constant altitude if possible. Since this heuristic assumes that the cost of moving from the current node to the goal is always along the cheapest path, or minimum of the $\kappa(\cdot)$ function, it is consistent, or never overestimates the remaining cost. Additionally, this heuristic monotonically decreases moving towards the goal, hence is admissible, and the path returned by the algorithm is guaranteed to be optimal [65, App. C].

The following example shows how the proposed heuristic determines optimal paths, Algorithm 5 has been used to plan a path near several buildings. The planner must plan a path from the start to a goal that is 398m away while staying safe distance from the buildings as much as possible. A plot of the weighting function can be seen in Figure 3.1, the parameters have been set to $\mu_1 = 50$, $\mu_2 = 10$, $\mu_3 = 1$, which has a maximum of 5.55 at $\alpha = 0$, and at $\alpha = 7$, the function has a value of 1.04, so the proximity to the wall is minimally penalized for nodes further than 7m from the wall. This $\kappa(\cdot)$ function is designed to ensure the UAV stays significantly far away from the buildings unless approaching the start or goal. Figure 3.2 shows the planned path using the proposed $A*$ planner, and the path successfully traverses from start to goal while staying safely away from the buildings. The sequences of waypoints between the different nodes is saved, and adjusted for wind by using the method in Section 3.1.2. △

3.1.2 Adjusting path plans for wind

A very important characteristic of UAVs is that the velocity of a UAV, captured through a global positioning system (GPS), is measured relative to the air and not the ground. As seen in Figure 3.3, wind can become a major factor when considering times to go between nodes while flying in windy conditions. While we do not account for the wind when optimizing

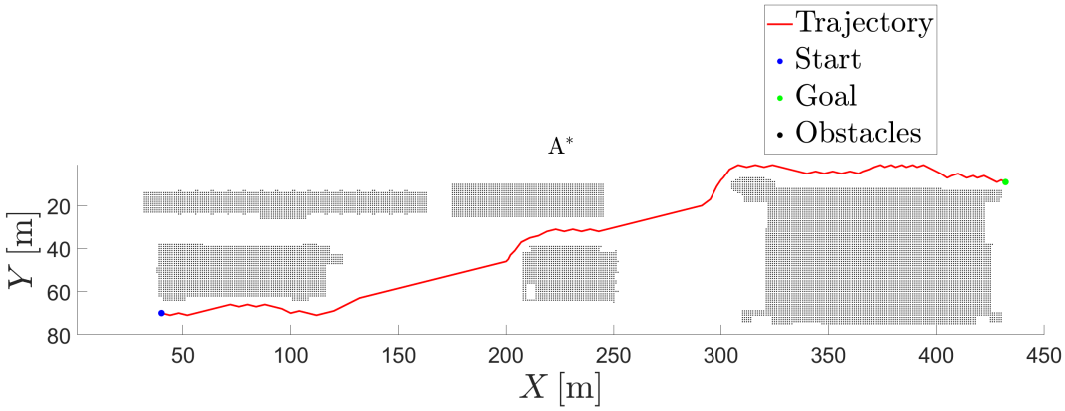

Figure 3.2: Plot of $A*$ path traversing between 5 buildings while maintaining safe distances from the surfaces.

the $A*$ paths, the change in flight time that may occur due to wind will be accounted for. In this work, it is assumed that the vehicle will always fly with its nose, or local $x(\cdot)$ axis, pointed towards its next waypoint. This is a common assumption for UAVs, since cameras or other sensors should point in the direction of motion for safety. The vehicle is also assumed to fly at a constant velocity relative to the air. This assumption is reasonable since in the proposed work, we are attempting to minimize the completion time of the mission, and the UAV should proceed as quickly as possible.

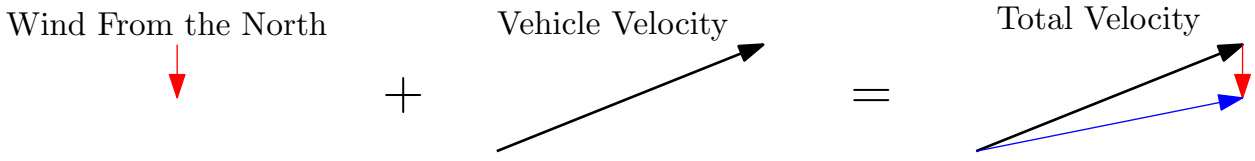

Figure 3.3: Relative Velocity Diagram. This figure shows how the introduction of wind affects the air-relative velocity of a flying vehicle. The vehicle velocity is assumed to be with respect to the air when there is no wind.

The linearized dynamics used for this method are given by,

$$\dot{x}(t) = v\cos(\psi(t)) + w_x, \qquad x(t_0) = x_0, \qquad t \geq t_0, \tag{3.5}$$

$$\dot{y}(t) = v\sin(\psi(t)) + w_y, \qquad y(t_0) = y_0, \tag{3.6}$$

where $x(\cdot)$ and $y(\cdot)$ denote the UAV's position in the global frame, an inertial frame fixed with the Earth, $v > 0$ denotes the velocity of the vehicle, assumed constant, $\psi(\cdot)$ denotes the heading angle, w_x, w_y denote the wind velocities in $x(\cdot)$ and $y(\cdot)$ assumed constant, respectively. The dynamics are modeled as a linearized version of UAV dynamics [66]. To ensure the vehicle moves to the next waypoint, the heading angle is solved as

$$\psi(t) \triangleq \tan^{-1}\left(\frac{x_{\text{next}} - x(t)}{y_{\text{next}} - y(t)}\right), \tag{3.7}$$

where $x_{\text{next}}, y_{\text{next}}$ denote the coordinates of the next waypoint along the $A*$ path, and

$$\tan^{-1}\frac{\alpha}{\beta} \triangleq \begin{cases} \tan^{-1}\dfrac{\alpha}{\beta}, & \beta > 0, \\ \tan^{-1}\dfrac{\alpha}{\beta} + \pi, & \alpha \geq 0, \beta < 0, \\ \tan^{-1}\dfrac{\alpha}{\beta} - \pi, & \alpha < 0, \beta < 0, \\ \pi/2, & \alpha > 0, \beta = 0, \\ -\pi/2, & \alpha < 0, \beta = 0, \\ 0, & \alpha = 0, \beta = 0, \end{cases} \tag{3.8}$$

denotes the *signed inverse tangent function*. To account for the wind, we find a new heading angle, denoted by $\psi_d(t)$, adjusted for wind. To ensure that the vehicle flies to the next waypoint, while pointing the nose of the vehicle in the direction of motion the following constraint must be satisfied,

$$\psi_d(t) \triangleq \tan^{-1}\left(\frac{v\cos(\psi(t)) + w_x}{v\sin(\psi(t)) + w_y}\right), \qquad t \geq 0. \tag{3.9}$$

For the rest of this chapter, we assume $\psi(t) = \psi_d(t)$. Since the velocity and heading angle are assumed constant between waypoints, the time to go between a waypoint (x_i, y_i) and

(x_{i+1}, y_{i+1}), for a consecutive pair of waypoints given by the $A*$ path, from some time $t_i \geq t_0$ when the vehicle departs to $t_{i+1} > t_1$ when the vehicle arrives at the second location is given by

$$t_{i+1} - t_i = \frac{x_{i+1} - x_i}{v\cos(\psi_d(t_i)) + w_x}, \tag{3.10}$$

or

$$t_{i+1} - t_i = \frac{y_{i+1} - y_i}{v\sin(\psi_d(t_i)) + w_y}, \tag{3.11}$$

and to calculate the total path time, the same calculation is performed over each pair of waypoints (x_i, y_i) to (x_{i+1}, y_{i+1}), $i \in \{0, \ldots, n_{\text{wp}} - 1\}$, where n_{wp} denotes the number of waypoints on the path. To ensure that Equation (3.10) is well-defined, it must be ensured that $v\cos(\psi_d(t_1)) + w_x \neq 0$, or that there is always some relative motion in the $x(\cdot)$ direction. A similar condition occurs for $y(\cdot)$ in Equation (3.11). This condition is ensured in this work by assuming a vehicle velocity relative to the air that is larger than the wind velocity. The found path times are used to construct the cost matrix for the optimization problem in Section 3.2.

3.1.3 Power modeling for UAVs based on momentum theory

In this section, a relationship is developed between the mass of the UAV and the power needed to keep the vehicle in hover. It has been shown experimentally, compared to constant velocity at speeds less than 10m/s, that hover can be a worst-case estimate of energy consumption for multi-rotor UAVs [52]. A simplified model of the power of a single rotor can be formulated using momentum theory. It is assumed that each rotor is a thin disk, the disturbed air of each disk is a cylindrical column, and air outside the column is undisturbed. The initial air velocity is assumed to be zero, and the flow is assumed to be irrotational [67,

Ch. 2]. The difference in air velocity is assumed to be related to the pressure difference using Bernoulli's equation, and it is assumed that the density of the flow is constant in any section of the air column, since the flow is incompressible. Following the same approach as in [68] and assuming the process has reached steady state, Bernoulli's equation for the outflow can then be written as,

$$p_i(t) + \Delta p(t) + \frac{1}{2}\rho v_i^2(t) = p_\infty(t) + \frac{1}{2}\rho v_\infty^2(t), \qquad t \geq t_0, \tag{3.12}$$

where $p_i(t)$ denotes the pressure at the disk on the inflow side, $\Delta p(t)$ denotes the change in pressure due to the acceleration of the flow, ρ denotes the air density, assumed constant, $v_i(t)$ denotes the velocity of the flow at the disk on the inflow side, $p_\infty(t)$ denotes the constant freestream pressure, and $v_\infty(t)$ denotes the velocity of the accelerated flow far behind the disk. The equation for the inflow is written as

$$p_\infty(t) = p_i(t) + \frac{1}{2}\rho v_i^2(t). \tag{3.13}$$

Substituting $p_\infty(t)$ into (3.12), it is shown that the change in pressure is

$$\Delta p(t) = \frac{1}{2}\rho v_\infty^2(t). \tag{3.14}$$

Next, conservation of momentum is applied to the disc. Let $f(t, m(t))$ denote the thrust force on the disc when is equal to

$$f(t) = \frac{\mathrm{d}m(t)}{\mathrm{d}t} v_\infty(t), \tag{3.15}$$

where $\frac{\mathrm{d}m(t)}{\mathrm{d}t}$ denotes the mass flow through the disc. Let the disc have a constant area denoted by A, and since the velocity on the inflow side of the disc is given by $v_i(t)$, the mass flow

can be written as $\frac{\mathrm{d}m(t)}{\mathrm{d}t} = \rho A v_i(t)$. Substituting $\frac{\mathrm{d}m(t)}{\mathrm{d}t}$ into (3.15), the thrust is given by,

$$f(t) = \rho A v_i(t) v_\infty(t). \tag{3.16}$$

As pressure is defined as a force per area, the change in pressure is equal to the thrust over the disk area, and from equations (3.14) and (3.16),

$$\Delta p(t) = \frac{1}{2}\rho v_\infty^2(t) = \frac{f(t)}{A} = v_i(t) v_\infty(t),$$

and by equating pressure with Equation (3.17) it can be seen that $v_\infty(t) = 2v_i(t)$ which gives a final relationship for thrust as,

$$f(t) = 2\rho A v_i^2(t). \tag{3.17}$$

This has shown that half of the velocity is generated above the disc, and the other half is generated below the disk. Solving for the induced velocity gives

$$v_i(t) = \sqrt{\frac{f(t)}{2\rho A}}. \tag{3.18}$$

The induced power of the ith rotor, denoted $P_i(t)$, is given by the force times the velocity, or the thrust times the induced velocity. Plugging in (3.18), the induced power is given by

$$P_i(m(t)) = f(t)\left(\sqrt{\frac{f(t)}{2\rho A}}\right) = \frac{f^{3/2}(t)}{\sqrt{2\rho A}}. \tag{3.19}$$

For a UAV in hover, the thrust is

$$f(t) = m(t) g, \tag{3.20}$$

where $g > 0$ denotes gravity and the mass is defined as $m(t) \triangleq m_v + m_p(t)$ where $m_v > 0$

denotes the unloaded mass of the vehicle, assumed constant, and the mass of the payload is denoted $m_p(t) \geq 0$. Assuming the n rotors carry the weight of the vehicle equally, the total power induced for a UAV can then be written as

$$P_{\text{total}}(m(t)) = n\frac{(m(t)g)^{3/2}}{\sqrt{2n^3\rho A}} = (m_v + m_p(t))^{3/2}\sqrt{\frac{g^3}{2n\rho A}}, \tag{3.21}$$

and it has been shown that a nonlinear relationship exists between the payload and the total induced power. To solve the routing problem described in Section 3.2, it is desirable to have a linear constraint. This can be done by safely assuming an upper bound of the estimate above. As an example, consider a quadrotor UAV, $n = 4$, with nominal mass $m_v = 3$kg,

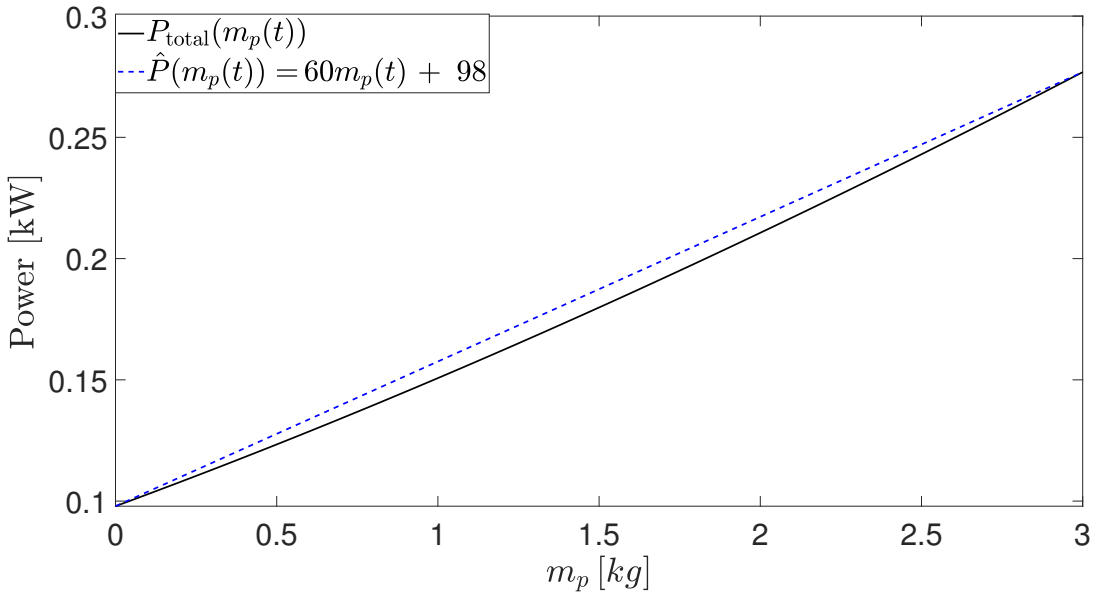

Figure 3.4: Plot of induced power, $P_{\text{total}}(m_p(t))$ versus the payload mass, $m_p(t)$ of a UAV. This plot shows how a simple linear upper bound of the induced power can be used to estimate energy consumption for a UAV which has a nominal mass of $m_v = 3$kg, and the payload is allowed to vary from 0 to 3kg. $\hat{P}(m_p(t))$ is the linear function that serves as an upper bound on the induced power.

that has 0.15m diameter propellers, $A = \frac{1}{4}\pi(0.15)^2$, and the payload can vary between 0 and 3kg. A plot, shown in Figure 3.4, shows the total induced power using (3.21). Furthermore, using the endpoints and the knowledge that the function is convex, an upper bound estimate

denoted, $\hat{P}(m_p(t))$, can be generated which is a linear function of the payload $m_p(t)$. Using the estimate of $\hat{P}(m_p(t))$, the payload transported between the waypoints, and the time to travel between the waypoints determined in Section 3.1.2, the energy to traverse between waypoints can be calculated and used to ensure that the battery is never depleted before reaching a refueling station or the depot. While the upper bound may be conservative if the payload has large variation, the ideal power induced calculated in (3.21) may be an underestimate of the actual power.

3.2 A split vehicle routing problem framework customized for UAVs

In the vehicle routing problem, a typical task for the vehicle is to deliver some payload to destinations considering the times at which the locations are visited. UAVs may have more tasks than simply delivering payload. For example, consider a scenario where a fleet of UAVs are contributing to a large-scale construction effort. There may be several different locations which need parts delivered. Additionally, in recent years UAVs have been used to update models of buildings under construction [69, 70, 71], and in a scenario such as this, the demand is not a delivery but a required time to perform scans of the area. In this section, an optimization model is developed such that a fleet of UAVs can complete a mission involving payload deliveries and required scans or surveillance where the ability of swapping batteries at known locations is allowed.

This problem is defined on a complete, directed network $G = (V, E)$ with nodes $V = \{0, 1, \cdots, n\}$, where $\{0\}$ denotes the depot or starting location, $n+1$ denotes the cardinality of V and includes the depot, the locations to visit, and the battery swap locations. For simplicity of notation in the modeling, let C_s denote the set of customers excluding the depot and the battery swap locations. E denotes the set of edges connecting the vertices,

and let $C_{ij}^k \geq 0$ denote the non-negative travel time to traverse the edge $(i,j) \in E$ for $i \in V, j \in V, i \neq j$. A key difference between the assumptions in Chapter 2 is that the travel times in this section are not assumed to be symmetric due to potential wind, that is c_{ij}^k is not necessarily equal to c_{ji}^k, for some arcs $(i,j),(j,i) \in E$ for the kth vehicle. The travel times do still satisfy the triangle inequality, that is, $C_{ij}^k \leq C_{il}^k + C_{lj}^k$, (i,j), (i,l), $(l,j) \in E$ for all vehicles. To service the customers, there are n_{veh} non-homogeneous vehicles with capacity $q_m > 0$. Since the vehicles non-homogeneous, we also track the cost matrix for each different vehicle, which can be different for UAVs due to different flight speeds. Each customer in C_s has a demand, and demands involving payload to be delivered are denoted $d_i \geq 0$, and demands that are time requirements for additional task are denoted $s_i \geq 0$. The demands differ from Chapter 2 in that the demands are no longer required to be less than vehicle capacity, and many UAVs may be required to service large demands due to small payload capacities. To include the possibility of swapping batteries, there are additional nodes in a refueling set, denoted N_r with the cardinality or $|N_r|$ equal to n_r. To consider the case that a vehicle may visit the same refuel site multiple times, n user-defined number of copies of the refuel locations are added to the list of nodes. In total, the list of all nodes including customers, the depot, all battery swap locations and their copies, is denoted N'.

In this chapter, a linear combination of the minimum cost and the minimum last service objective are considered, since we deduced in Section 2.5 that using just the last service time objective could make the total cost extremely high. The minimum last service is the maximum time at which any vehicle is still performing a task at any of the locations. This is important since in this work, deliveries are not considered as instantaneous and surveillance or monitoring tasks can incur non-negligible time requirements. The minimum cost objective is defined as

$$z_c \triangleq \sum_{k=1}^{m} \sum_{(i,j)\in A} C_{ij}^k x_{ij}^k, \tag{3.22}$$

where x_{ij}^k denotes the main decision variable, binary, and equal to 1 if vehicle k travels along

the (i, j) arc, and is 0 otherwise, and m is the number of vehicles. The minimum last service time, z_{ls}, is defined as,

$$z_{ls} \triangleq \max_{k=\{1,\ldots,n_{\text{veh}}\}} \left[\sum_{(i,j)\in E: j\neq 0} C_{ij}^k x_{ij}^k + \sum_{r=1}^{n_s} \sum_{j\in N} p_{j,r}^k x_{ij}^k \right], \tag{3.23}$$

where $p_{j,r}^k$ denotes the servicing times at node j for vehicle k, the subscript r denotes which type of service the vehicle is performing, and n_s denotes the number of service types.

The objective function for this mixed integer linear program is given by

$$\min \quad \lambda_1 z_c + \lambda_2 z_{ls}, \tag{3.24}$$

where $\lambda_1, \lambda_2 > 0$ are weights for the objective functions given by (3.22) and (3.23). This formulation allows to use a lexicographic combination of the two objective functions studied in Chapter 2. It is known that with a minimum last delivery-type of objective function, there can be many optimal solutions. By adding a very small $0 < \lambda_1 << 1$, we try to find the optimal solutions that have the minimum cost.

The constraints used to ensure route connectivity and vehicle usage are given by,

$$\sum_{k=1}^{n_v} \sum_{j\in N'} x_{ij}^k \geq 1, \quad i \in C_s, \tag{3.25}$$

$$\sum_{k=1}^{n_v} \sum_{j\in N'} x_{ij}^k \leq 1, \quad i \in N_r \setminus \{0\}, \tag{3.26}$$

$$\sum_{i\in N'} x_{ij}^k = \sum_{i\in N'} x_{ji}^k, \quad j \in N', \quad k = \{1, \ldots, n_{\text{veh}}\}, \tag{3.27}$$

$$\sum_{j\in N'\setminus\{0\}} x_{0j}^k = 1, \qquad k = \{1, \ldots, n_{\text{veh}}\}, \tag{3.28}$$

where constraint (3.25) ensures that all customers are visited by at least one vehicle. Constraint (3.26) states that all refuel or battery swap locations will be visited by at most one

vehicle. The extra copies of each location allow for a vehicle to visit the same battery swap location more than once, but by using copies instead of allowing the same arc to be used multiple times, the size of the state space is reduced. Constraint (3.27) is the standard flow conservation constraint which states that a vehicle arriving to node j must leave node j.

The payload delivery constraints are captured by,

$$y_{ij}^k \leq q_k x_{ij}^k, \quad i \in N', \qquad j \in N', \quad i \neq j, \quad k = \{1, \ldots, n_{\text{veh}}\}, \tag{3.29}$$

$$\sum_{j \in N'} \sum_{k=1}^{n_{\text{veh}}} y_{ij}^k - \sum_{j \in N'} \sum_{k=1}^{n_{\text{veh}}} y_{ji}^k = d_j, \quad j \in C_s, \tag{3.30}$$

$$\sum_{i \in N' \setminus \{0\}} y_{ij}^k \geq \sum_{i \in N' \setminus \{0\}} y_{ji}^k, \quad j \in N', \quad k = \{1, \ldots, n_{\text{veh}}\}, \tag{3.31}$$

where $y_{ij}^k \geq 0$ denotes the units of payload carried by vehicle k , along the arc (i, j), $k \in \{1, \ldots, n_v\}$. Constraint (3.29) ensures that the capacity of the vehicle, denoted by $q > 0$, is never exceeded along an arc. This constraint also ensures that a vehicle must visit a node to make a delivery to that node. In this work, the capacity of the vehicles is not assumed to be uniform, and the payload can very depending on the vehicle. Constraint (3.30) ensures that the delivery demands are met for each customer. Since split deliveries are allowed in this formulation, constraint (3.31) is added to ensure that a vehicle cannot leave a node with more payload than it entered with, thus excluding payload transfers.

The constraints used to track the times at which each node is visited are given by,

$$0 \leq \tau_0^k, \quad k \in \{1, \ldots, n_{\text{veh}}\}, \tag{3.32}$$

$$\tau_j^k \geq \tau_i^k + \left(C_{ij}^k + \left(\sum_{r=1}^{n_s} p_{jr}^k \right) \right) x_{ij}^k - T x_{ij}^k,$$
$$i \in N', \quad j \in N' \setminus \{0\}, \quad i \neq j, \quad k \in \{1, \ldots, n_{\text{veh}}\}, \tag{3.33}$$

where τ_j^k for $j \in N'$ denotes the mission time at which vehicle k visits node j. Constraint

(3.32) provides the lower bound as the initial time, assumed 0. Constraint (3.33) tracks the time at which node j is visited by vehicle k. This is the time at which the vehicle arrives at the node, not when service is complete. The variable $T >> 0$ is used to help eliminate sub-tours using the Miller-Tucker-Zemlin formulation [72].

The battery remaining for vehicle k is denoted by b_k, and this is tracked along the routes through the following constraints,

$$b_j^k \leq b_i^k - C_{ij}^k f(y_{ij}^k) x_{ij}^k + B(1 - x_{ij}^k), \quad i \in N', \quad j \in C_s, \quad i \neq j, \tag{3.34}$$

$$b_j = B, \quad j \in R' + \{0\}, \qquad k \in \{1, \ldots, n_{\text{veh}}\}, \tag{3.35}$$

$$b_j^k \geq \min(f(y_{ij}^k) C_{j0}^k, f(y_{ij}^k)(C_{jl}^k + C_{l0}^k)), \quad l \in R, \quad j \in C_s, \quad k \in \{1, \ldots, n_{\text{veh}}\}. \tag{3.36}$$

Constraint (3.34) tracks the remaining endurance of the kth vehicle at node j, the effects of carrying heavier payloads is captured by the function $f(y_{ij}^k)$, and the maximum battery capacity is given by $B > 0$. Using the same approach as in Section 3.1.3, this becomes a linear function where m_v in equation (3.21) is the known vehicle mass and m_p becomes y_{ij}^k. Since energy is considered as power times time, we multiply the power function $f(y_{ij}^k)$ by the cost matrix which captures the time to go between nodes. Constraint (3.35) captures that by visiting refuel locations, the battery is charged to maximum. Constraint (3.36) ensures that visiting the next node and getting to either the depot or a refuel station is feasible.

In cases where time-based tasks are required and not just delivery tasks, the variable p_{jr}^k can become an optimization variable. If a service time s_j is required at node $j \in C$, then we have the following constraint,

$$\sum_{k=1}^{m} p_{jr}^k = s_{jr}, \quad j \in N' \setminus \{0\}, \quad r \in \{1, \ldots, n_s\}. \tag{3.37}$$

In cases where a vehicle is tasked with dropping off a payload and the service time is assumed constant, then p_{jr}^k can be considered as a constant value at each node visited by vehicle k.

3.3 Numerical simulations

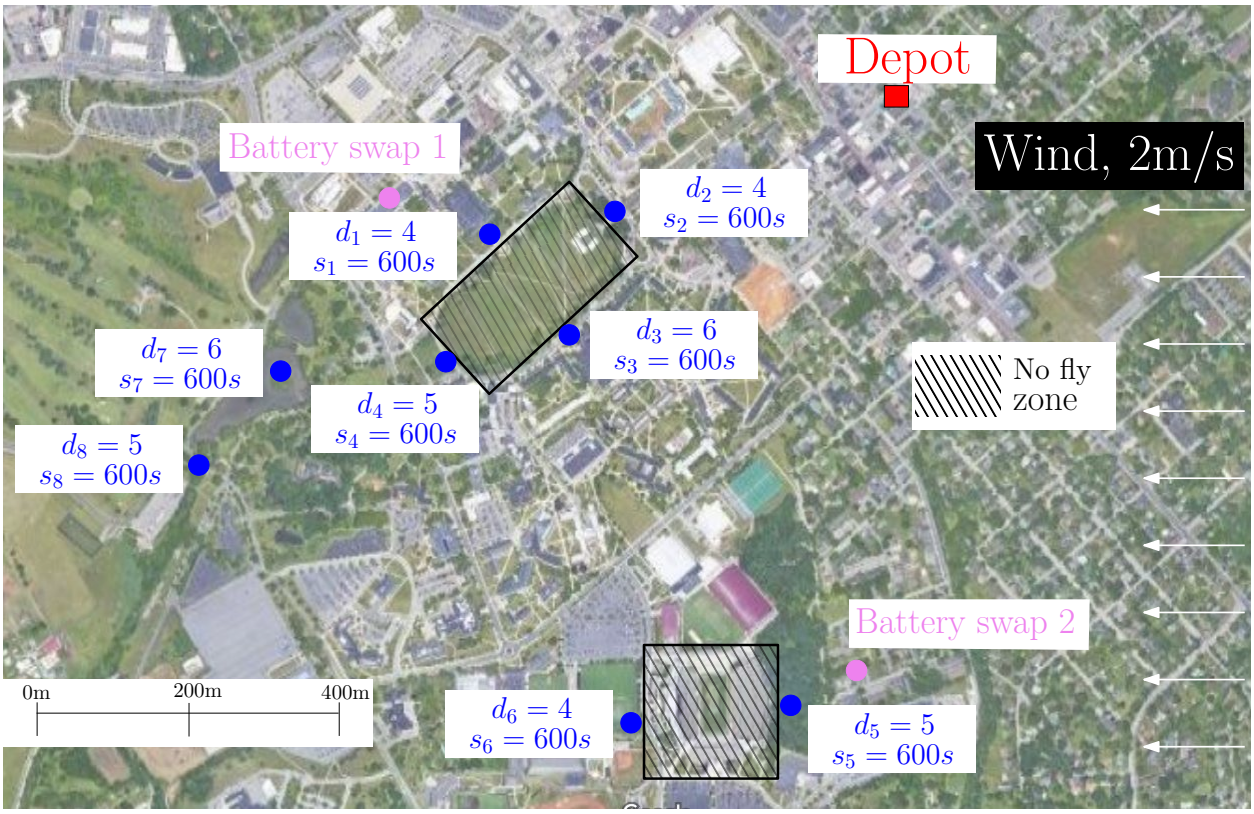

Figure 3.5: Locations of the depot shown in red, all locations to visit and their demands are shown in blue, and the battery swap locations shown in purple. The black marked areas and their interiors are considered as no fly zones for safety purposes.

In this section, a sample scenario is provided which utilizes a combination of the techniques presented in this chapter. Consider the following scenario based around the campus of Virginia Tech where there are 8 locations of interest near 4 areas, the drillfield, the football stadium, the drone park, and the duck pond. At the edge of each of these locations, food can be delivered. The orders of food are 0.5kg each, and the locations have a demand of 2kg to 3.5kg food in increments of 0.5kg, and the food is coming from a rooftop restaurant on Main street. There are two locations where the UAVs can land to swap batteries, the roof of Derring Hall and the VT police station near the football stadium. In addition to delivery demands, it is assumed that events are occurring and some of the UAVs will be used to take images and video of each location. For the safety of the large gatherings, the

airspace above both the drillfield and the stadium will be considered off limits. Using freely available online tools, the GPS coordinates of all of the locations are taken and converted to position coordinates. In this problem, altitude is being ignored as the vehicles are assumed to fly above buildings, the ascend and descend times are accounted for in service time, and the path planner will avoid the no fly zones with lateral maneuvers.

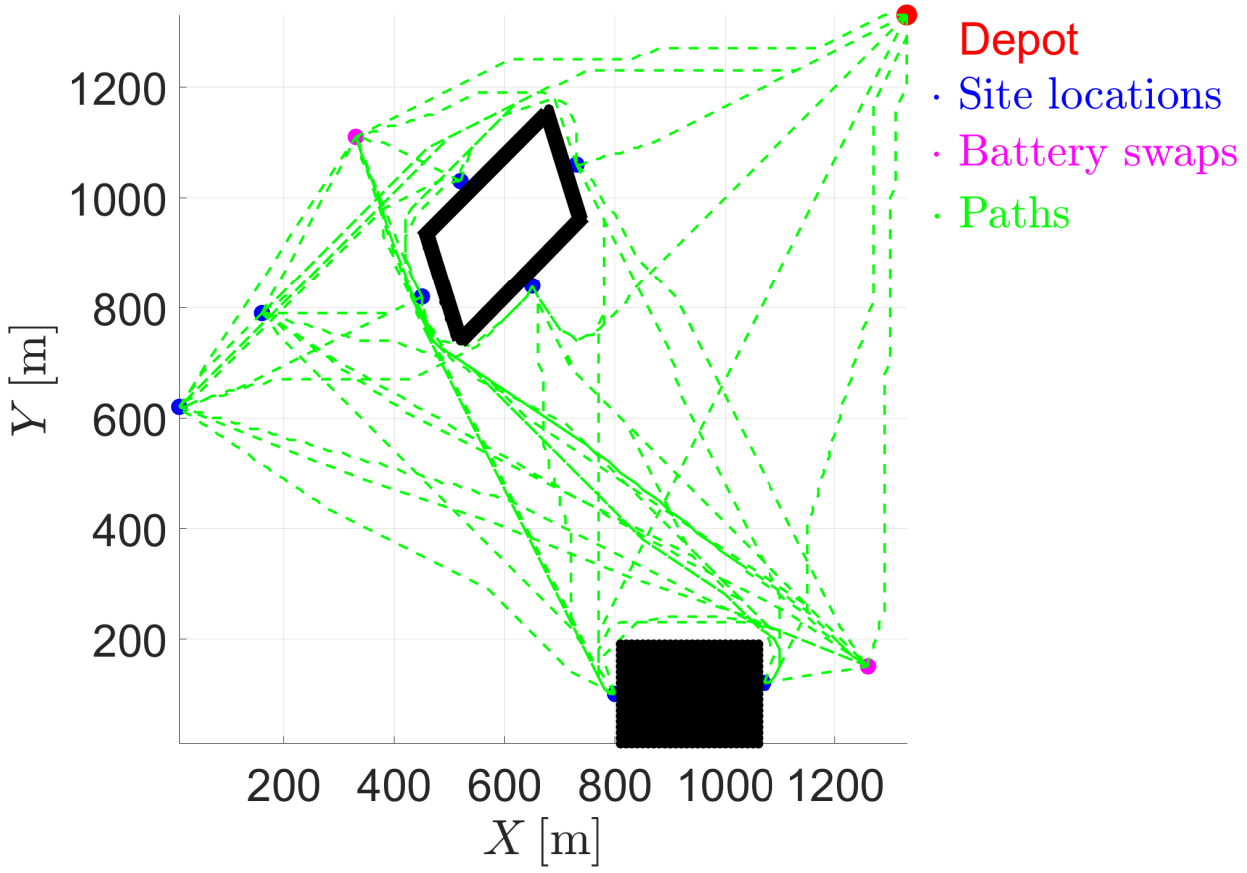

Figure 3.6: $A*$ paths of routing problem example. The paths are shown in green, the locations of the depot are shown in red, the customers are shown in blue, and the battery swap locations shown in purple. The black marked areas and their interiors are considered as no fly zones for safety purposes. It can be seen that the custom heuristic pushes the flight paths away from the borders of the no fly zones for increased safety.

For this sample scenario, there is one copy of the battery swap locations, $N' = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}$, $C = 8$, $n_r = 2$ The customer demands are given by, $d = [4, 4, 6, 5, 5, 4, 6, 5]$, and the ascend and descend times are captured by the service times $p_j = 60$, $j \in N' \setminus \{0\}$, where it is assumed that it takes 60s at each location to to descend

and ascend. At each customer, at least 10 minutes or 600 seconds worth of images or video should be taken for archive purposes, $s_j = 600$, $j \in C_s$. The area has been discretized to squares of 10m by 10m for the purposes of path planning. Figure 3.5 shows the locations of the depot, the customers and their demands, and the battery swap locations. Also shown is the no fly zones which are captured by the black-colored bubbles and the interiors.

To generate the possible set of paths for the problem, the $A*$ algorithm presented in section 3.1.1 is used. The values used in the heuristic are $\mu_1 = 50$, $\mu_2 = 10$, $\mu_3 = 2$, which has a maximum of 5.55 at $\alpha = 0$, and at $\alpha = 2$, the function has a value of 1.04, so the proximity to the wall is minimally penalized for nodes further than, 2 units, or 20m from the wall. For this problem, there will be two different UAV types available. The first type will be UAVs equipped with high resolution cameras for performing the desired image and video taking. There are three of this vehicle type, and they have a maximum endurance of 20 minutes, $B = 1200$s, with an assumed flight speed between locations of 10m/s. The second type is for delivering the food, and four of them are available. These vehicles have a nominal weight of $m_2 = 5kg$ with a maximum payload capacity of $m_p = 5$kg, and the maximum endurance with no payload is 15 minutes, $B = 900$s. Using 4 propellers with a radius of 0.15m, plotting equation (3.21) and finding a linear upper bound, we have that the power is given by $P(M) = 1.04M + 1.62$ where $M = m_2 + m_p$, which is used as $f(\cdot)$ in (3.34) and (3.36). The paths are updated to account for a 2m/s wind from the East direction, or X axis using the vector algebra method in section 3.1.2. The cost matrix for the first vehicle type is given, in seconds of travel time for no payload. Looking at the time to go from the depot to node 2, $c_{02}^{\text{type 1}} = 95$s, compared to going from node 2 to the depot, $c_{20}^{\text{type 1}} = 213$s, the effects of wind can be seen. This matches intuition in that moving from the depot to node 2 to the depot is moving West, or along the direction of wind giving the UAV a velocity relative to the ground that is faster than its nominal velocity relative to the air. For brevity, this cost matrix does not include the extra rows and columns due to the copies of the refuel locations

are omitted, but they are captured by appropriately copying the values in the table.

$$C^{\text{type 1}} = \begin{bmatrix} 0 & 131 & 95 & 166 & 175 & 211 & 237 & 199 & 239 & 141 & 192 \\ 263 & 0 & 60 & 93 & 44 & 211 & 183 & 71 & 111 & 30 & 216 \\ 213 & 93 & 0 & 81 & 106 & 178 & 159 & 131 & 170 & 73 & 184 \\ 299 & 147 & 103 & 0 & 54 & 152 & 128 & 82 & 101 & 107 & 160 \\ 331 & 63 & 163 & 89 & 0 & 167 & 139 & 38 & 70 & 53 & 172 \\ 231 & 371 & 208 & 222 & 308 & 0 & 78 & 184 & 180 & 218 & 25 \\ 282 & 230 & 167 & 141 & 167 & 124 & 0 & 163 & 148 & 192 & 84 \\ 408 & 145 & 240 & 166 & 80 & 372 & 285 & 0 & 39 & 63 & 192 \\ 478 & 215 & 310 & 206 & 157 & 379 & 310 & 67 & 0 & 103 & 183 \\ 301 & 66 & 134 & 149 & 60 & 347 & 224 & 77 & 139 & 0 & 227 \\ 197 & 418 & 242 & 275 & 355 & 50 & 162 & 434 & 400 & 418 & 0 \end{bmatrix}.$$

To test the formulation, the problem data and the optimization formulation (3.24)-(3.37) are programmed in Python and solved using the Gurobi solver. Setting $\lambda_1 = 1$ and $\lambda_2 = 0$, simulations are performed while minimizing the total time for the mission. The total cost is 9,674s and the last service is completed at 3,720s. For this objective function, we get the paths as in figure 3.7 for the first vehicle type gathering video footage. All of the type 1 UAVs have to swap batteries at least one time, and the optimal routes are listed in Table 3.1. The second type of UAVs visit nodes, as shown in Figure 3.8 as a Matlab plot and

Table 3.1: Routes and costs for type 1 vehicles under total cost objective.

Vehicle	Route	Surveillance
1	$0 \to 1 \to 9 \to 4 \to 7 \to 6 \to 0$	[600,0,304.5,379,600]
2	$0 \to 2 \to 0$	[600]
3	$0 \to 8 \to 7 \to 9 \to 4 \to 3 \to 10 \to 5 \to 0$	[0,221,0,295.5,600,0,600]

Figure 3.9 where the paths are overlaid on a Google satellite view of the area. The optimal paths are given in Table 3.2.

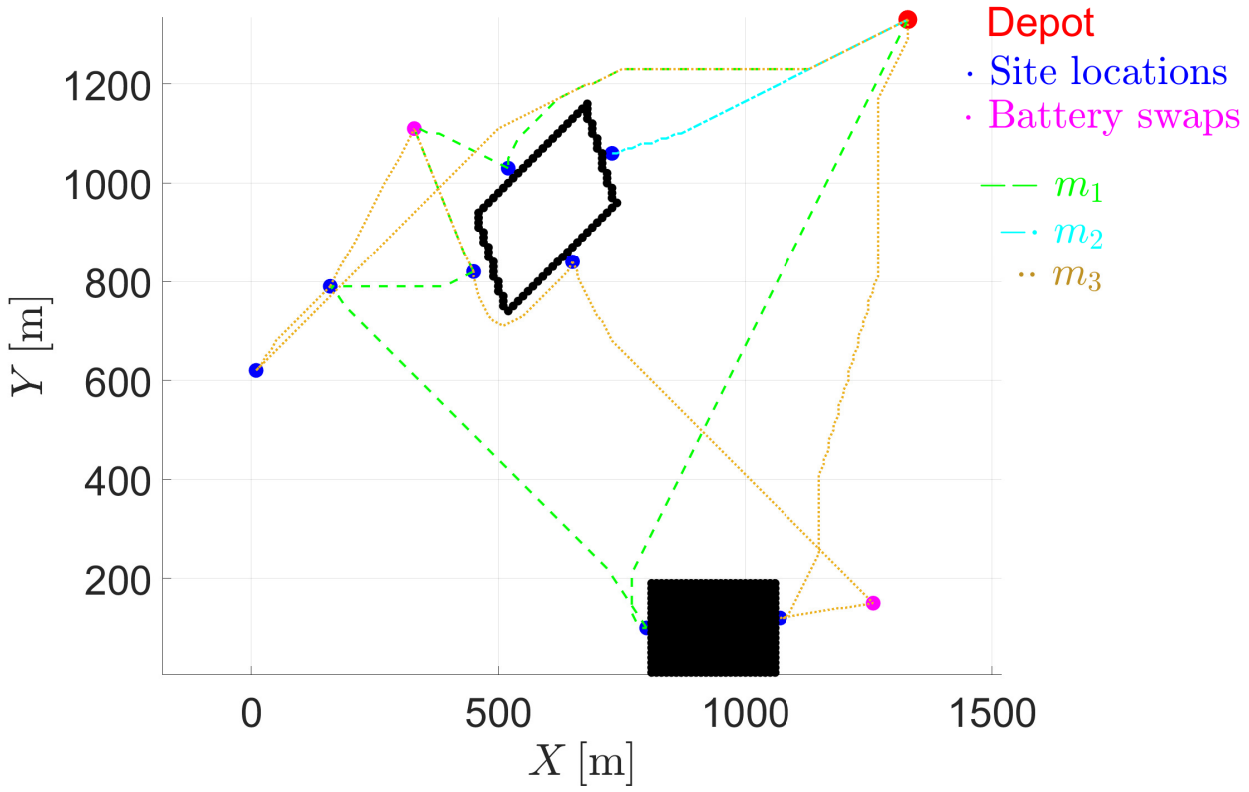

Figure 3.7: The optimal paths of the type 1 UAVs which are tasked with gathering video footage at the locations while minimizing the total mission time.

Table 3.2: Routes and costs for type 2 vehicles under total cost objective.

Vehicle	Route	Delivery
4	$0 \to 5 \to 6 \to 10 \to 0$	[4,5]
5	$0 \to 11 \to 4 \to 8 \to 9 \to 0$	[0,5,5,0]
6	$0 \to 1 \to 7 \to 9 \to 0$	[4,6,0]
7	$0 \to 2 \to 3 \to 9 \to 0$	[4,6,0]

The simulation is re performed by setting $\lambda_1 = 0.001$ and $\lambda_2 = 1$ to minimize the last service time. By setting $0 < \lambda_1 << 1$, an attempt is made to find the best minimum cost within the optimal last service time. In this case, the total cost is determined as 10,070s with a last service time of 2,128s. This is a 43% improvement in last delivery time with an increase of 16% in total mission time further validating the ideas of Chapter 2 that the objective functions can heavily influence the outcomes. In this case, the type 1 vehicles have optimal tours given in Table 3.3. The second vehicle type can actually perform the exact same minimum cost tours as the previous case. This is due to the fact that the image

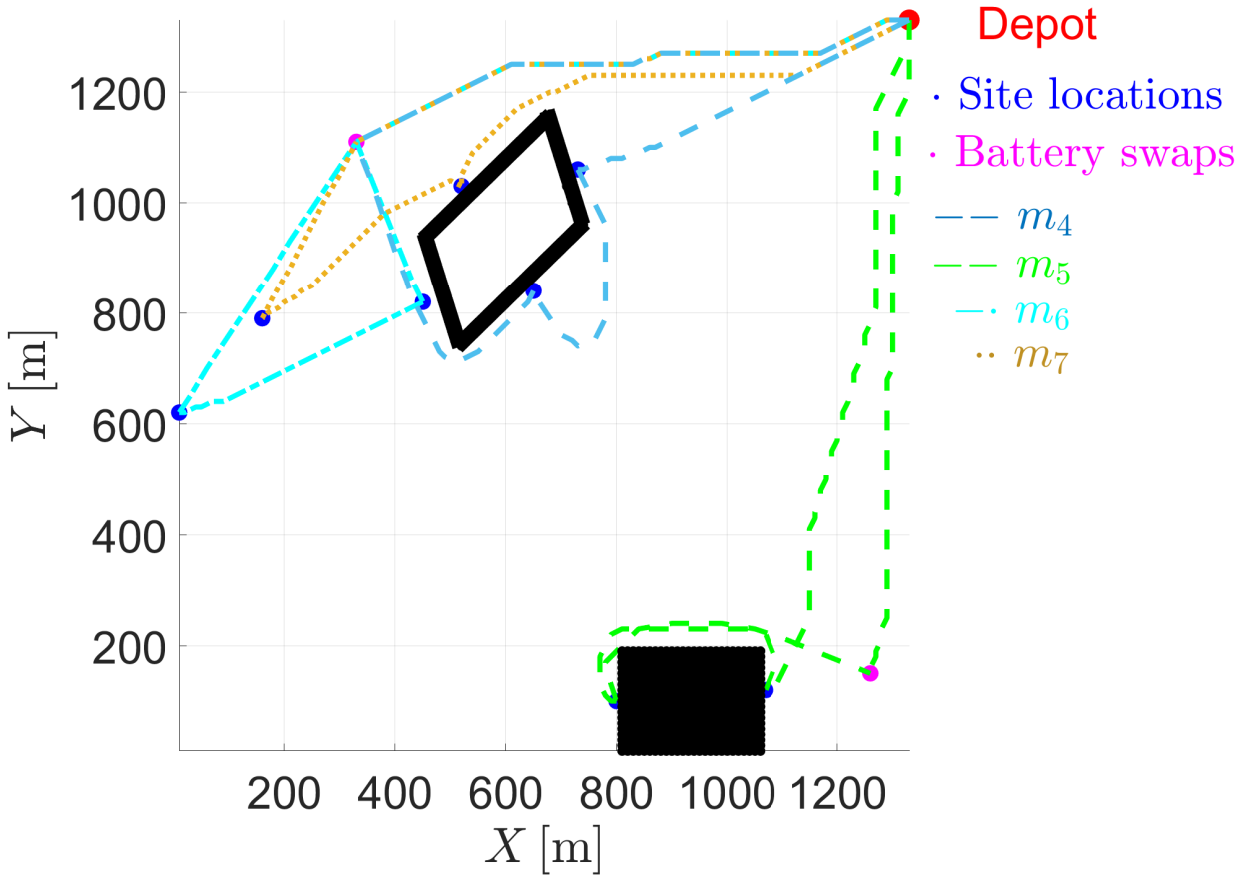

Figure 3.8: The optimal paths of the type 2 UAVs which are tasked with delivering food to the customers while minimizing total cost. The plots are overlaid with a Google maps image of the area to show the viability of the approach.

gathering mission is much more time consuming than the actual payload deliveries. Figure 3.10 shows the routes of the type 1 vehicles while minimizing last service time.

The simulations were performed in 4 additional scenarios, no wind, 2m/s wind from the west, 2m/s wind from the north, and 2m/s wind from the south. The results are shown in table 3.4, and it can be seen that the wind causes in increase in objective function values in most cases. The one case in which wind improved the objective function is when the wind is blowing to the west and the goal is to minimize the last service time. In this case, the UAVs get to utilize the wind to move West a faster velocity, since the depot is at the eastern-most part of the grid. In this case, the last service time is 6% faster than the case with no wind. However, in other cases, the wind typically causes the objective values to increase. For the

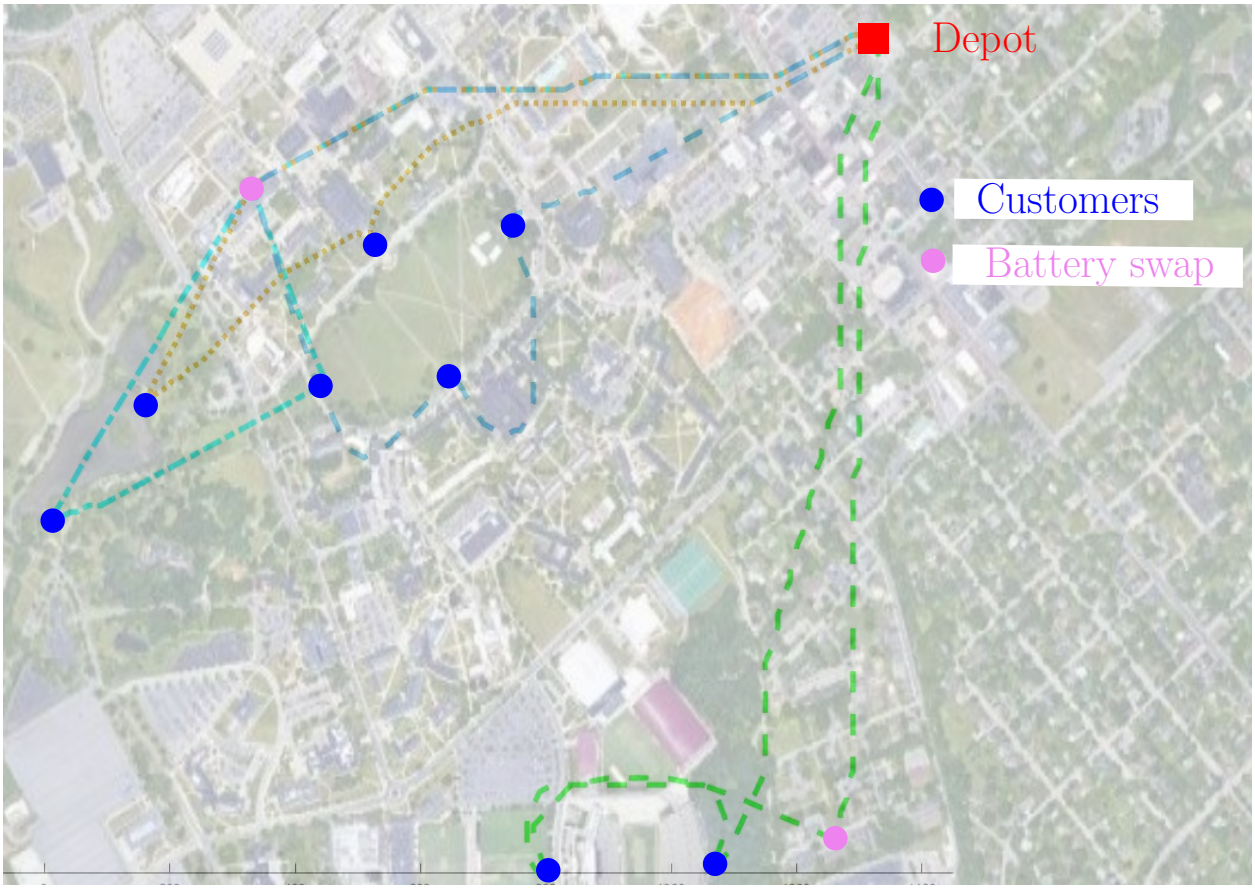

Figure 3.9: The optimal paths of the type 2 UAVs which are tasked with delivering food to the customers while minimizing total cost. The plots are overlaid with a google maps image of the area to show the viability of the approach.

minimum cost objective, the average increase is around 10%. A potential downside to the method presented in this work is that the wind is not accounted for in the path planner currently. By including the wind vector in calculations of the cost in $A*$, potentially faster paths could be found, thus reducing the total cost objective in the optimization problem.

3.4 Conclusion

In this chapter, a method is presented to make the vehicle routing problem more suitable for UAVs. An $A*$ path planning technique with a custom heuristic is utilized to generate collision-free paths that maintain a safe distance from buildings or surfaces. Assuming a

Table 3.3: Routes and costs for type 1 vehicles under last delivery objective.

Vehicle	Route	Surveillance
1	$0 \to 1 \to 11 \to 2 \to 9 \to 4 \to 0$	[600,0,341,0,600]
2	$0 \to 6 \to 10 \to 5 \to 3 \to 2 \to 9 \to 7 \to 0$	[600,0,198,322,437,0,173]
3	$0 \to 5 \to 10 \to 3 \to 2 \to 9 \to 7 \to 0$	[402,0,278,259,0,427]

Table 3.4: UAV routing in different wind scenarios. The notation is given where, $c(c)$, denotes the total cost, in seconds, of the solution that minimizes cost, $ls(c)$ denotes the last service time, in seconds, of the solution that minimizes cost, $c(ls)$ denotes the cost of the solution, in seconds, that minimizes last service time, and $ls(ls)$ denotes the last service time, in seconds, of the solution which minimizes the last service time.

Wind	$c(c)$	$ls(c)$	$c(ls)$	$ls(ls)$
None	9,641	3,891	10,040	2,273
2m/s east	9,674	3,720	10,070	2,128
2m/s west	10,614	4,082	11,349	2,641
2m/s north	10,671	4,377	10,923	2,393
2m/s south	10,662	3,862	10,779	2,462

constant vehicle and wind velocity, a method is presented to account for wind by adjusting the times it takes to traverse the paths generated by $A*$. These wind-adjusted path times underlie the cost matrix of an optimization problem that considers UAVs in a vehicle routing problem with the additional capability of visiting refuel or battery swap locations to extend their range. Furthermore, we consider the possibility that all demands are not of the payload delivery type, and using UAVs for mapping or surveillance scenarios is possible. An example is presented where two sets of UAVs are tasked with both delivering food and taking video footage of an event with wind present, and where the limited range promotes the necessity of the UAVs to refuel.

Both this chapter and the previous chapter have considered the problem of routing vehicles, specifically UAVs, for missions involving payload delivery. Since payloads can cause unknowns in the UAV's dynamics, the next chapter will focus on how a control law can be designed for unknown nonlinear systems such as UAVs where also constraints can be set a priori by the user on the tracking error and control inputs to ensure safety of the vehicle and persons nearby.

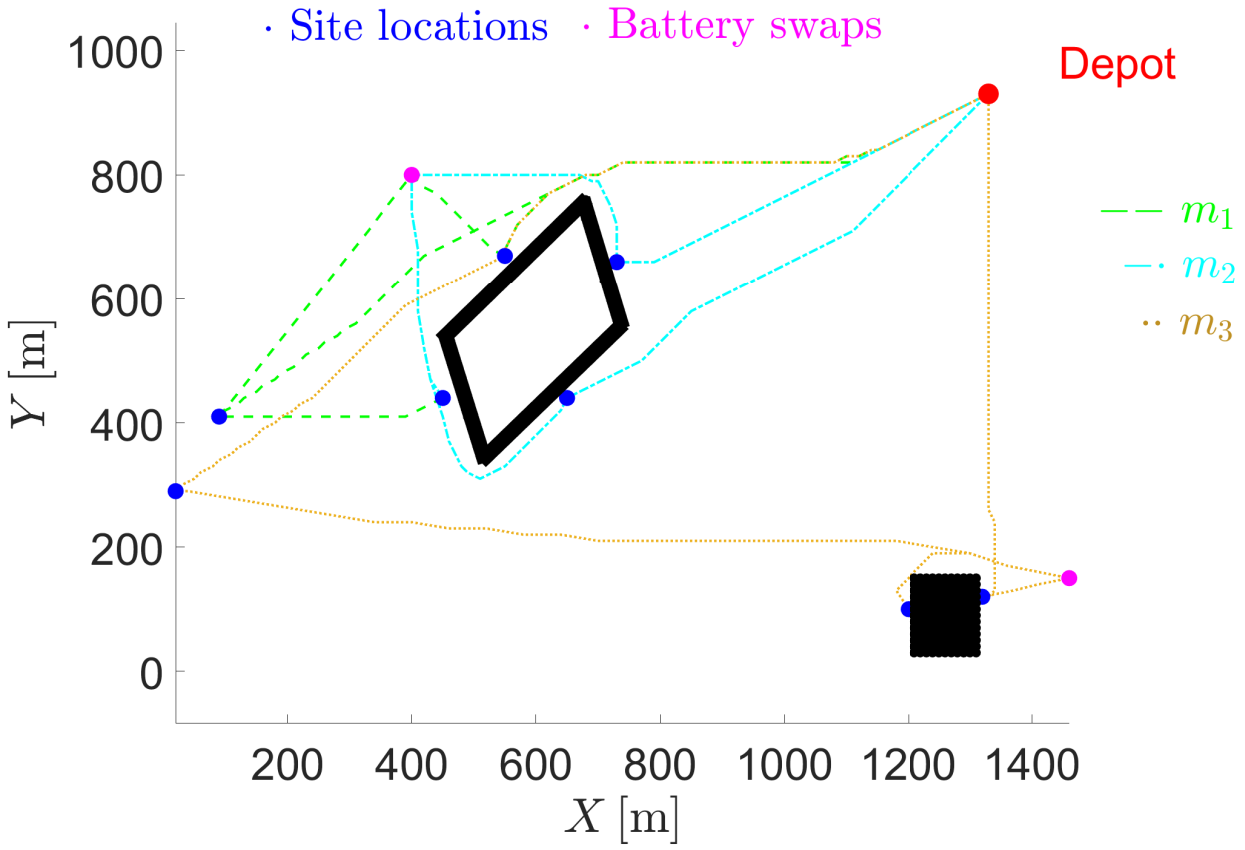

Figure 3.10: The optimal paths of the type 1 UAVs which are tasked with gathering video footage at the locations while minimizing last service time.

Chapter 4

Novel model reference adaptive control laws for improved transient dynamics and guaranteed saturation constraints

Chapters 2 and 3 focused on generating optimal paths to route UAVs for scenarios including payload deliveries. Both this chapter and Chapter 5 5 will focus on developing control laws to enable the UAVs to complete the challenging tasks proposed by the routing problem. An unknown or non-static payload can induce changes in both the mass and inertial properties of a UAV. These changes may cause coupling between the translational and rotational equations of motion by shifting the center of gravity, further challenging the control laws [73]. The control approach in this thesis is broken into two main categories. The first, discussed in this chapter, considers the UAVs with parametric uncertainties that could be performing mapping or delivery tasks which do not include contacting any surfaces. Secondly, in Chapter 5, a control law is developed and tested for UAVs that have discontinuous dynamics, such as those that come into contact with hard surfaces.

In previous work, we have shown that model reference adaptive control (MRAC) is a very successful control technique for dealing with uncertainties in the system, specifically when carrying and releasing payloads [74]. In classical MRAC, the adaptive rates must be

tuned to meet multiple competing objectives. Large adaptive rates guarantee rapid convergence of the trajectory tracking error to zero. However, large adaptive rates may also induce saturation of the actuators, especially in cases of small UAVs that typically have a limited power capability. Furthermore, excessive overshoots of the closed-loop system's trajectory tracking error are common with high adaptive rates, and this is an extremely undesirable behavior for systems that are in operation near persons or buildings . Conversely, low adaptive rates may produce unsatisfactory trajectory tracking performances. To overcome these limitations, in the classical MRAC framework, the adaptive rates must be tuned through an iterative process. Alternative approaches require to modify the plant's reference model or the reference command input. In this chapter, we present the first MRAC laws for nonlinear dynamical systems affected by matched and parametric uncertainties that constrain both the closed-loop system's trajectory tracking error and the control input at all times within user-defined bounds, and enforce a user-defined rate of convergence on the trajectory tracking error. By applying the proposed MRAC laws, the adaptive rates can be set arbitrarily large and both the plant's reference model and the reference command input can be chosen arbitrarily. The user-defined rate of convergence of the closed-loop plant's trajectory is enforced by introducing a user-defined auxiliary reference model, which converges to the trajectory tracking error obtained by applying the classical MRAC laws before its transient dynamics has decayed, and steering the trajectory tracking error to the auxiliary reference model at a rate of convergence that is higher than the rate of convergence of the plant's reference model. The ability of the proposed MRAC laws to prescribe the performance of the closed-loop system's trajectory tracking error and control input is guaranteed by barrier Lyapunov functions. Numerical simulations of a delta wing aircraft illustrate both the applicability of our theoretical results and their effectiveness compared to other techniques such as prescribed performance control, which allows to constrain both the rate of convergence and the maximum overshoot on the trajectory tracking error of uncertain systems.

4.1 Literature review

In the presence of matched and parametric uncertainties, classical MRAC guarantees both uniform asymptotic convergence of the closed-loop system's trajectory tracking error and boundedness of both the trajectory tracking error and the adaptive gains [75, Ch. 9]. Several MRAC laws have been proposed to guarantee uniform ultimate boundedness of the closed-loop system's trajectory tracking error in the presence of unmatched uncertainties [75, Ch. 10],[76, 77, 78]. Because of its relative ease of implementation, MRAC has been applied to solve control problems in multiple fields, including aerospace [75], biomedical [79], mechanical [80], and electrical engineering [81], to name a few.

The structure of a classical MRAC law is fixed and, given a reference model, the user can tune the adaptive rates to meet some design specifications on the closed-loop system's performance. These specifications include, for example, guaranteeing high responsiveness of the feedback control law to large trajectory tracking errors, bounding the maximum overshoot of the trajectory tracking error, and constraining the $\mathcal{L}_\infty$-norm of the control input. Large adaptive rates guarantee high responsiveness of the feedback control law to large trajectory tracking errors. However, in the transient phase, large adaptive rates may induce large and rapid excursions of both the control inputs and the trajectory tracking error [75, p. 389]. Both the trajectory tracking error's maximum overshoot and the $\mathcal{L}_\infty$-norm of the control input can only be estimated in a conservative manner [82, 83], and adaptive rates can only be tuned *a posteriori* through an iterative process to prevent both the closed-loop plant's trajectory from exceeding its operative range and the control input from saturating the plant's actuators. As an alternative to tuning the adaptive rates, the user could tune the zeroth-order term in the underlying Lyapunov equation. However, it is not common practice to meet user-defined levels of performance of the closed-loop system by tuning this parameter, since direct correlations between the the zeroth-order term in the underlying Lyapunov equation, the trajectory tracking error, and the control input have not been found

yet [84]. The structure of classical MRAC does not allow to tune the rate of convergence of the closed-loop system's trajectory tracking error and impose that the closed-loop plant's trajectory reaches the reference model's trajectory before the reference model's transient dynamics has decayed [75, pp. 389-390].

The main results of this chapter are *MRAC laws for prescribed performance*. These MRAC laws guarantee asymptotic convergence to zero of the trajectory tracking error and uniform boundedness of both the adaptive gains and the trajectory tracking error. These MRAC laws also allow to impose *a priori* a user-defined rate of convergence on the closed-loop system's trajectory tracking error, and enforce user-defined constraints on both the trajectory tracking error and the control input at all times. Remarkably, the proposed MRAC laws allow the user to set the adaptive rates arbitrarily large and choose the reference command input arbitrarily, and do not require to modify the reference model. To the authors' best knowledge, this is the first result of this kind within the MRAC framework. The absence of an MRAC law that combines all these properties has been highlighted in several recent works such as [82, 85, 86].

The adaptive control framework that most closely matches the features of the proposed MRAC laws is *prescribed performance control* [87, 88]. This technique applies to uncertain affine in the control dynamical systems and allows to impose both a user-defined rate of convergence and a user-defined bound on the maximum overshoot of the closed-loop system's trajectory tracking error by transforming the constrained system into an unconstrained one. Although prescribed performance control applies to a larger class of dynamical systems than the proposed MRAC laws, it is unable to explicitly impose user-defined constraints on the control input, which is a distinctive feature of the proposed MRAC laws. Furthermore, the proposed MRAC laws for prescribed performance allow to impose not only the maximum overshoot of the closed-loop system's trajectory tracking error, but a larger class of constraints on the trajectory tracking error.

The proposed MRAC laws for prescribed performance are direct adaptive laws, since the adaptive gains are not enforced to converge to the corresponding unknown, true values [75, Ch. 7]. Indirect or mixed direct-indirect adaptive controls, that is, control techniques that require convergence of the adaptive gains to their respective true values, are useful in problems in which the user needs to characterize online the plant's dynamics or identify faults and failures. These parameter estimation features are achievable by inducing persistence of excitation of the control input at all times [89, Ch. 6], augmenting the regressor vector [90, 91, 92], including a model prediction error in the adaptive laws as in *mixed direct-indirect MRAC* [93, 94], or employing some parameter identification technique such as adaptive observers [95] or least-squares estimation methods [96, Ch. 6], [97, Ch. 4]. Although the proposed MRAC laws are not suitable for online parameter estimation, they guarantee user-defined levels of performance at all times despite modeling uncertainties, do not require persistence of excitation of the control input, do not require to augment the regressor vector, and do not involve observers or estimators.

The proposed MRAC laws for prescribed performance are attained in three consecutive steps. Firstly, we modify the classical MRAC law by augmenting the vector of feedback variables with a user-defined vector function. Similarly to the classical MRAC law, the proposed adaptive control law guarantees uniform boundedness of the closed-loop system's trajectory tracking error, uniform boundedness of the adaptive gains, and uniform asymptotic convergence of the closed-loop system's trajectory tracking error. However, a difference with the classical MRAC law is that the user is now able to arbitrarily choose part of the feedback control law and meet additional design specifications. For instance, the authors of [86] recently applied this approach to create an anti-windup system for the MRAC framework.

As a second step toward the proposed MRAC laws for prescribed performance, we introduce a new MRAC framework that we named *two-layer MRAC*. Applying the two-layer MRAC framework, the user introduces an auxiliary reference model in addition to the plant's reference model that characterizes the classical MRAC framework. This auxiliary reference

model is designed to converge to the closed-loop system's trajectory tracking error obtained by applying the classical MRAC framework before the transient dynamics of the plant's reference model has decayed, and the two-layer MRAC law steers the closed-loop system's trajectory tracking error to the auxiliary reference model at a rate of convergence that is higher than the rate of convergence of the plant's reference model. Thus, two-layer MRAC guarantees convergence of the closed-loop plant's trajectory to the plant's reference model before its transient dynamics has decayed.

Several approaches have been proposed to set explicitly the rate of convergence of the trajectory tracking error. These approaches modify the reference model's dynamics by adding an error feedback term [98, 99, 100], use barrier Lyapunov functions [82, 83, 101], or involve estimators of the plant's nonlinearities [84, 102, 103, 104]. Adding an error feedback term in the reference model's dynamics alters the original reference model and deprives the users of their prerogative of designing arbitrarily the reference model and the reference trajectory. Using barrier Lyapunov functions to bound the trajectory tracking error, the norm of the control input grows asymptotically as the trajectory tracking error approaches the boundary of the constraint set. Finally, designing estimators of the plant nonlinearities may result in a complex task for problems of practical interest. Two-layer MRAC allows to impose a user-defined rate of convergence to the closed-loop plant's trajectory without any constraints on the adaptive rates, without modifying the user-defined reference model's dynamics, without employing barrier Lyapunov functions, and without using estimators of the plant's nonlinearities. Worthy of mention are *concurrent learning MRAC* [90, 91, 92] and its extension *hybrid MRAC for unmatched uncertainties* [105, 106]. By appending recorded data of output and state estimate vectors to the baseline adaptive law, these techniques guarantee convergence of the adaptive gains to their ideal values without requiring persistence of excitation of the input functions, and guarantee faster convergence of the closed-loop system's trajectory tracking error than by applying classical MRAC [107]. However, neither concurrent learning MRAC nor hybrid MRAC for unmatched uncertainties allow to set the closed-loop system's

rate of convergence explicitly. Furthermore, both approaches require the use of observers that, as already discussed, are not needed by two-layer MRAC.

As a third step toward the proposed MRAC laws for prescribed performance, we present an original MRAC law that allows to constrain both the trajectory tracking error and the control input at all time. This result is achieved by employing barrier Lyapunov functions to enforce constraints on both the closed-loop system's trajectory tracking error and the control input and hence, to guarantee that the constraints on the trajectory tracking error do not necessarily imply larger control efforts. Also this MRAC law comprises a user-defined vector function, which is eventually used to implement our original two-layer MRAC framework, enforce a user-defined rate of convergence on the closed-loop plant's trajectory, and achieve the proposed MRAC laws for prescribed performance.

Existing results on the synthesis of MRAC laws that account explicitly for saturation constraints involve linear dynamical systems [108, 109, 110, 111, 112] or feedback-linearizable plants [113, 114]. In some cases, such as [112], also the reference model is partly modified to prevent saturation of the control input. Remarkably, the proposed MRAC laws for prescribed performance allow to set arbitrarily large adaptive gains, do not require to modify the reference model, and apply to nonlinear plants that are not necessarily feedback-linearizable. More frequently, constraints on the control input are enforced implicitly by employing the projection operator [115, 116] or barrier Lyapunov functions to bound the adaptive gains [83]. However, if the adaptive gains are bounded within user-defined constraint sets, then uniform ultimate boundedness of the closed-loop system's trajectory tracking error is guaranteed, but not its asymptotic convergence [82].

Numerical examples show the applicability of our original results. Special emphasis is given to comparing the performance of the proposed MRAC laws for prescribed performance to the performance of classical prescribed performance control [87, 88]. Our simulations highlight how both the proposed MRAC laws and classical prescribed performance control

are able to impose a user-defined rate of convergence on the closed-loop plant's trajectory and a user-defined maximum overshoot on the closed-loop system's trajectory tracking error. However, despite the proposed MRAC laws, classical prescribed performance control does not verify the user-defined saturation constraints.

4.2 Notation, definitions, and mathematical preliminaries

In this section, we establish notation, definitions, and review some basic results. Throughout this chapter, we use two types of mathematical statements, namely existential and universal statements. *Existential statements* are in the form: the condition C is verified for some $x \in \mathcal{X}$. *Universal statements* are in the form: the condition C holds for all $x \in \mathcal{X}$; for universal statements, we generally omit the words "for all" and write: C holds, $x \in \mathcal{X}$. If the universal statements $C_1, \ldots, C_n$ hold over the same set $\mathcal{X}$, then we write: C_1 holds, $x \in \mathcal{X}$, C_2 holds, ..., and C_n holds.

Let $\mathbb{N}$ denote the *set of positive integers*, $\mathbb{Z}$ denote the *set of integers*, $\mathbb{R}$ denote the *set of real numbers*, $\mathbb{C}$ the *set of complex numbers*, $\mathbb{R}^n$ the *set of* $n \times 1$ *real column vectors*, $\mathbb{R}^{n\times m}$ the *set of* $n \times m$ *real matrices*, and $\mathcal{B}_\varepsilon(x)$ the *open ball centered* at $x \in \mathbb{R}^n$ with *radius* ε. The *interior of the set* $\mathcal{E} \subset \mathbb{R}^n$ is denoted by $\mathring{\mathcal{E}}$ and the *boundary of* $\mathcal{E}$ is denoted by $\partial\mathcal{E}$. The *real part of* $z \in \mathbb{C}$ is denoted by $\mathrm{Re}(z)$. The *zero vector* in $\mathbb{R}^n$ is denoted by 0_n or 0, the *zero matrix* in $\mathbb{R}^{n\times m}$ is denoted by $0_{n\times m}$ or 0, and the *identity matrix* in $\mathbb{R}^{n\times n}$ is denoted by I_n or I. The block-diagonal matrix formed by $M_i \in \mathbb{R}^{n_i\times n_i}$, $i = 1, \ldots, p$, is denoted by $M = \mathrm{blockdiag}\,(M_1, \ldots, M_p)$. The *transpose of* $B \in \mathbb{R}^{n\times m}$ is denoted by B^{T}, the *trace of* $A \in \mathbb{R}^{n\times n}$ is denoted by $\mathrm{tr}(A)$, the *eigenvalues of* A *with maximum and minimum real parts* are denoted by $\lambda_{\max}(A)$ and $\lambda_{\min}(A)$, respectively. We write $\|\cdot\|$ for the *Euclidean vector norm* and the corresponding *equi-induced matrix norm*, and $\|\cdot\|_{\mathrm{F}}$ for the *Frobenius matrix*

norm [117, pp. 18-19].

Next, we recall the notion of rate of convergence [118] of asymptotically stable, linear, time-invariant dynamical systems on $\mathbb{R}^n$, whose equilibrium point is at the origin [117, Def. 2.47].

Definition 4.1. Let $x : [t_0, \infty) \to \mathbb{R}^n$ denote the flow of a linear, time-invariant, uncontrolled, asymptotically stable dynamical system on $\mathbb{R}^n$, whose equilibrium point is at the origin. Then, the *rate of convergence of* $x(\cdot)$ is defined as

$$\alpha_{\max}(x(\cdot)) \triangleq \sup_{x_0 \in \mathbb{R}^n \setminus \{0\}} \liminf_{t \to \infty} \frac{\log \|x(t)\|}{t_0 - t}. \tag{4.1}$$

It follows from Definition 4.1 that the rate of convergence of a linear dynamical system is the supremum over all $\alpha > 0$ such that $\lim_{t\to\infty} e^{\alpha t}\|x(t)\| = 0$ [118, p. 55]. Consider the linear, time-invariant, controlled dynamical system

$$\dot{x}(t) = Ax(t) + Bu(t), \qquad x(t_0) = x_0, \qquad t \geq t_0, \tag{4.2}$$

where $x(t) \in \mathbb{R}^n$, $t \geq t_0$, $u(t) \in \mathbb{R}^m$ is bounded, $A \in \mathbb{R}^{n\times n}$ is Hurwitz, and $B \in \mathbb{R}^{n\times m}$. The next result provides both the rate of convergence of (4.2) and an upper bound on the solution of (4.2).

Proposition 4.2. *Consider the linear, time-invariant, dynamical system* (4.2)*. It holds that* $\alpha_{\max}(x(\cdot)) = -\mathrm{Re}(\lambda_{\max}(A))$ *and*

$$\|x(t)\| \leq \gamma_1 \|x_0\| e^{\xi(t)(t-t_0)} - \frac{\gamma_1 \|B\|}{\gamma_2 \mathrm{Re}(\lambda_{\max}(A))} \sup_{t \in [t_0, \infty)} \|u(t)\|, \qquad t \geq t_0, \tag{4.3}$$

where $\xi : [t_0, \infty) \to \mathbb{R}$, $\lim_{t\to\infty} (\xi(t) - \mathrm{Re}(\lambda_{\max}(A))) = 0^+$, $\gamma_1 \geq 1$, *and* $\gamma_2 \in (0, 1)$.

Proof: The solution of (4.2) is given by $x(t) = e^{A(t-t_0)}x_0 + \int_{t_0}^{t} e^{A(t-\tau)}Bu(\tau)\mathrm{d}\tau$, $t \geq t_0$ [119, p. 33]. Thus, applying the triangle inequality and the Cauchy-Schwarz inequality, it

holds that

$$\|x(t)\| \leq \|e^{A(t-t_0)}\|\|x_0\| + \|B\| \sup_{t\in[t_0,\infty)} \|u(t)\| \int_{t_0}^{t} \left\|e^{A(t-\tau)}\right\| \mathrm{d}\tau, \qquad t \geq t_0. \tag{4.4}$$

Now, it follows from [120] that $\|e^{A(t-t_0)}\| \leq \gamma_1 e^{\xi(t)(t-t_0)}$, $t \geq t_0$, where

$$\lim_{t\to\infty} \left(\xi(t) - \mathrm{Re}(\lambda_{\max}(A))\right) = 0^+, \tag{4.5}$$

and hence, setting $u(t) = 0$, it follows from (4.1) that $\alpha_{\max}(x(\cdot)) = -\mathrm{Re}(\lambda_{\max}(A))$. Furthermore, since $\|I_n\| = 1$, it follows from Fact 11.18.8 of [121] that $\|e^{A(t-\tau)}\| \leq \gamma_1 e^{\gamma_2 \mathrm{Re}(\lambda_{\max}(A))(t-\tau)}$ for all $t \in [t_0, \infty)$ and for all $\tau \in [t_0, t]$, where $\gamma_1 \geq 1$ and $\gamma_2 \in (0, 1)$. Thus, (4.3) follows from (4.4) and the fact that

$$\int_{t_0}^{t} e^{\gamma_2 \mathrm{Re}(\lambda_{\max}(A))(t-\tau)} \mathrm{d}\tau \leq \int_{t_0}^{\infty} e^{\gamma_2 \mathrm{Re}(\lambda_{\max}(A))(t-\tau)} \mathrm{d}\tau = -\left(\gamma_2 \mathrm{Re}(\lambda_{\max}(A))\right)^{-1}, \qquad t \geq t_0.$$

■

4.3 Motivations for a novel model reference adaptive control framework

In order to motivate this work, it is worthwhile to recall the classical formulation of the MRAC architecture and highlight some of its critical aspects. Specifically, consider the nonlinear *plant's dynamics*

$$\dot{x}(t) = Ax(t) + B\Lambda \left[u(t) + \Theta^{\mathrm{T}} \Phi(t, x(t))\right], \qquad x(t_0) = x_0, \qquad t \geq t_0, \tag{4.6}$$

where $x(t) \in \mathcal{D} \subseteq \mathbb{R}^n$, $t \geq t_0 \geq 0$, denotes the *plant's trajectory*, $0 \in \mathcal{D}$, $u(t) \in \mathbb{R}^m$ denotes the *control input*, $A \in \mathbb{R}^{n\times n}$ is unknown, $B \in \mathbb{R}^{n\times m}$ is known, $\Lambda \in \mathbb{R}^{m\times m}$ is diagonal, positive-

definite, and unknown, $\Theta \in \mathbb{R}^{N\times m}$ is unknown, the *regressor vector* $\Phi : [t_0, \infty) \times \mathcal{D} \to \mathbb{R}^N$ is jointly continuous in its arguments, and $\Phi(t, \cdot)$ is locally Lipschitz continuous in x uniformly in t on compact subsets of $[t_0, \infty)$. We assume that the pair $(A, B\Lambda)$ is controllable; in problems of practical interest, where the entries of A and Λ are unknown, this hypothesis can be verified since the structure of A is usually known [75, p. 281]. Both A and Λ capture *parametric uncertainties*, and $\Theta^{\mathrm{T}}\Phi(t, x)$, $(t, x) \in [t_0, \infty)\times\mathcal{D}$, captures *matched uncertainties*; the choice of the regressor vector $\Phi(\cdot, \cdot)$ to parameterize uncertainties is usually based on some prior knowledge of the plant dynamics [75, Ch. 9]. Without loss of generality, it is possible to formulate the classical MRAC framework assuming that B in (4.6) is unknown and $\Lambda = I_m$. However, considering B as a known matrix and Λ as an unknown diagonal matrix allows to capture faults and failures in each control channel separately.

Consider also the *plant's reference model*

$$\dot{x}_{\mathrm{ref}}(t) = A_{\mathrm{ref}}x_{\mathrm{ref}}(t) + B_{\mathrm{ref}}r(t), \qquad x_{\mathrm{ref}}(t_0) = x_{\mathrm{ref},0}, \qquad t \geq t_0, \tag{4.7}$$

where $x_{\mathrm{ref}}(t) \in \mathbb{R}^n$, $t \geq t_0$, denotes the *reference trajectory*, the *reference command input* $r(t) \in \mathbb{R}^m$ is continuous and bounded, $A_{\mathrm{ref}} \in \mathbb{R}^{n\times n}$ is Hurwitz, and $B_{\mathrm{ref}} \in \mathbb{R}^{n\times m}$, and assume there exist $(K_x, K_r) \in \mathbb{R}^{n\times m} \times \mathbb{R}^{m\times m}$ such that the *matching conditions*

$$A_{\mathrm{ref}} = A + B\Lambda K_x^{\mathrm{T}}, \tag{4.8}$$

$$B_{\mathrm{ref}} = B\Lambda K_r^{\mathrm{T}}, \tag{4.9}$$

are verified. Since A_{ref}, B, and B_{ref} are known and the structures of A and Λ are known, in numerous problems of practical interest it is possible to prove the existence of K_x and K_r that verify the matching conditions (4.8) and (4.9), respectively [75, p. 282]. To appreciate the generality of the classical MRAC framework, it is worthwhile to recall that for all multi-body mechanical systems affected by modeling uncertainties, there exist baseline controllers

that exploit the plant's passivity properties, reduce the plant's equations of motion to the same form as (4.6), and verify the matching conditions (4.8) and (4.9) [122, Ch. 8], [123].

It follows from (4.6)–(4.9) that

$$\dot{e}(t) = A_{\text{ref}}e(t) + B\Lambda\left[u(t) - K_x^{\mathrm{T}}x(t) - K_r^{\mathrm{T}}r(t) + \Theta^{\mathrm{T}}\Phi(t, x(t))\right], \quad e(t_0) = x_0 - x_{\text{ref},0}, \quad t \geq t_0, \tag{4.10}$$

where $e(t) \triangleq x(t) - x_{\text{ref}}(t)$ denotes the *trajectory tracking error*. Therefore, the matching conditions imply that if A, Λ, and Θ were known and

$$u(t) = K_x^{\mathrm{T}}x(t) + K_r^{\mathrm{T}}r(t) - \Theta^{\mathrm{T}}\Phi(t, x(t)), \qquad t \geq t_0, \tag{4.11}$$

then $\lim_{t\to\infty} e(t) = 0$ [75, pp. 281-282].

Consider the symmetric, positive-definite solution $P \in \mathbb{R}^{n\times n}$ of the *Lyapunov equation*

$$0 = A_{\text{ref}}^{\mathrm{T}}P + PA_{\text{ref}} + Q, \tag{4.12}$$

where the zeroth-order term $Q \in \mathbb{R}^{n\times n}$ is a user-defined symmetric, positive-definite matrix, and consider the *feedback control law*

$$\phi(\pi(t, x, r), \hat{K}) = \hat{K}^{\mathrm{T}}\pi(t, x, r), \qquad (t, x, r, \hat{K}) \in [t_0, \infty) \times \mathcal{D} \times \mathbb{R}^m \times \mathbb{R}^{(n+m+N)\times m}, \tag{4.13}$$

where the *adaptive gain matrix* $\hat{K}(\cdot)$ verifies the *adaptive law*

$$\dot{\hat{K}}(t) = -\Gamma\pi(t, x(t), r(t))e^{\mathrm{T}}(t)PB, \qquad \hat{K}(t_0) = \hat{K}_0, \qquad t \geq t_0, \tag{4.14}$$

the user-defined *adaptive rate matrices* $\Gamma_x \in \mathbb{R}^{n\times n}$, $\Gamma_r \in \mathbb{R}^{m\times m}$, and $\Gamma_\Theta \in \mathbb{R}^{N\times N}$ are

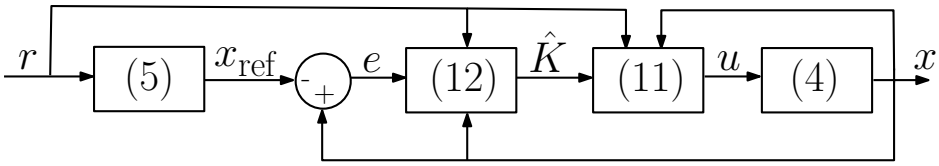

Figure 4.1: Schematic representation of the classical MRAC architecture to steer the state $x(\cdot)$ of the nonlinear plant (4.6) toward the state $x_{\text{ref}}(\cdot)$ of the plant's reference model (4.7). Applying the control law (4.13) and the adaptive law (4.14), Theorem 4.3 guarantees uniform boundedness of both the closed-loop plant's trajectory error $e(\cdot)$ and the adaptive gain matrix $\hat{K}(\cdot)$. Moreover, Theorem 4.3 guarantees uniform asymptotic convergence of $e(\cdot)$ to zero.

symmetric and positive-definite, $\Gamma \triangleq \text{blockdiag}\,(\Gamma_x, \Gamma_r, \Gamma_\Theta)$, and

$$\pi(t, x, r) \triangleq \left[x^{\mathrm{T}}, r^{\mathrm{T}}, -\Phi^{\mathrm{T}}(t, x)\right]^{\mathrm{T}}, \qquad (t, x, r) \in [t_0, \infty) \times \mathcal{D} \times \mathbb{R}^m, \tag{4.15}$$

denotes the *vector of feedback variables*. If $u(t) = \phi(\pi(t, x(t), r(t)), \hat{K}(t))$, $t \geq t_0$, then it follows from (4.6)–(4.9) that the *closed-loop trajectory error dynamics* is given by

$$\dot{e}(t) = A_{\text{ref}} e(t) + B\Lambda\Delta K^{\mathrm{T}}(t)\pi(t), \qquad e(t_0) = x_0 - x_{\text{ref},0}, \qquad t \geq t_0, \tag{4.16}$$

where $\Delta K(t) \triangleq \hat{K}(t) - K$ and $K \triangleq \left[K_x^{\mathrm{T}}, K_r^{\mathrm{T}}, \Theta^{\mathrm{T}}\right]^{\mathrm{T}}$. The next result illustrates the classical MRAC architecture.

Theorem 4.3 ([75, Th. 9.2])**.** *Consider the plant's dynamics* (4.6)*, the plant's reference model* (4.7)*, the feedback control law* (4.13)*, and the adaptive law* (4.14)*. If the matching conditions* (4.8) *and* (4.9) *are verified, then both the trajectory tracking error* $e(\cdot)$ *and the adaptive gain matrix* $\hat{K}(\cdot)$ *are uniformly bounded, and* $e(t) \to 0$ *as* $t \to \infty$ *uniformly in* $t_0 \in [0, \infty)$*.*

Theorem 4.3 proves that applying the feedback control law (4.13) and the adaptive law (4.14), the closed-loop plant's trajectory $x(\cdot)$ converges asymptotically to the reference trajectory $x_{\text{ref}}(\cdot)$. Figure 4.1 provides a schematic representation of the MRAC control architecture outlined by Theorem 4.3.

In the following, we highlight two critical aspects of the classical MRAC framework. Specifically, it follows from (4.16), (4.7), and Proposition 4.2 that $\alpha_{\max}(e(\cdot)) = -\mathrm{Re}(\lambda_{\max}(A_{\mathrm{ref}}))$ and $\alpha_{\max}(x_{\mathrm{ref}}(\cdot)) = -\mathrm{Re}(\lambda_{\max}(A_{\mathrm{ref}}))$. However, it would be desirable to design an MRAC law so that

$$\alpha_{\max}\left(e(\cdot)\right) \geq \alpha, \tag{4.17}$$

where $\alpha > -\mathrm{Re}(\lambda_{\min}(A_{\mathrm{ref}})) \geq -\mathrm{Re}(\lambda_{\max}(A_{\mathrm{ref}}))$ is a user-defined parameter [75, pp. 389-390]. The two-layer MRAC proposed in this chapter allows to guarantee this design specification.

Classical MRAC is affected by an additional limitation. Specifically, it is common practice to choose the adaptive rate matrix Γ in (4.14) so that $\mathrm{Re}(\lambda_{\min}(\Gamma))$ is large and hence, the adaptive gain $\hat{K}(\cdot)$ responds rapidly to large values of the norm of the tracking error $\|e(\cdot)\|$. However, it follows from (4.13) that large values of $\|\hat{K}(\cdot)\|$ may imply larger values of the norm of $\|u(\cdot)\|$ and hence, the plant's actuators may saturate in the presence of large trajectory tracking errors. Moreover, it follows from (4.16) that larger values of $\|\hat{K}(\cdot)\|$ may produce larger values of $\|e(\cdot)\|$ and hence, the plant's trajectory may exceed its operative range. In this chapter, a novel MRAC framework is provided to constrain both the control input and the trajectory tracking error at all time, while the adaptive rates are set arbitrarily large.

4.4 A novel model reference adaptive control law

In this section, a variation of classical MRAC is presented, wherein the vector of feedback variables is augmented by a vector function that can be designed to meet user-defined design specifications, in addition to the uniform boundedness of the adaptive gains and the uniform asymptotic convergence of the trajectory tracking error dynamics guaranteed by Theorem

4.3. For the statement of the next theorem, consider the feedback control law

$$\phi(\pi(t,x,r),g(t,x,e,r),\hat{K},\hat{K}_g)=\hat{K}^{\mathrm{T}}\pi(t,x,r)+\hat{K}_g^{\mathrm{T}}g(t,x,e,r),$$
$$(t,x,e,r,\hat{K},\hat{K}_g)\in[t_0,\infty)\times\mathcal{D}\times\mathbb{R}^n\times\mathbb{R}^m\times\mathbb{R}^{(n+m+N)\times m}\times\mathbb{R}^{p\times m},\tag{4.18}$$

where the vector of feedback variables $\pi(\cdot,\cdot,\cdot)$ is given by (4.15), $g:[t_0,\infty)\times\mathcal{D}\times\mathbb{R}^n\times\mathbb{R}^m\to\mathbb{R}^p$ is jointly continuous in its arguments, $g(t,\cdot,\cdot,r)$ is locally Lipschitz continuous in (x,e) uniformly in (t,r) on compact subsets of $[t_0,\infty)\times\mathbb{R}^m$, $g(\cdot,x,e,r)$ is bounded for all $t\in[t_0,\infty)$, $\hat{K}(\cdot)$ verifies the adaptive law (4.14), $\hat{K}_g(\cdot)$ verifies the adaptive law

$$\dot{\hat{K}}_g(t)=-\Gamma_g g(t,x(t),e(t),r(t))e^{\mathrm{T}}(t)PB,\qquad \hat{K}_g(t_0)=\hat{K}_{g,0},\qquad t\geq t_0,\tag{4.19}$$

the user-defined adaptive rate matrix $\Gamma_g\in\mathbb{R}^{p\times p}$ is symmetric and positive-definite, P denotes the symmetric, positive-definite solution of (4.12), $x(\cdot)$ denotes the solution of (4.6) with

$$u(t)=\phi(\pi(t,x(t),r(t)),g(t,x(t),e(t),r(t)),\hat{K}(t),\hat{K}_g(t)),\tag{4.20}$$

$e(\cdot)$ denotes the solution of

$$\dot{e}(t)=A_{\mathrm{ref}}e(t)+B\Lambda\left[\Delta K^{\mathrm{T}}(t)\pi(t,x(t),r(t))+\hat{K}_g^{\mathrm{T}}(t)g(t,x(t),e(t),r(t))\right],$$
$$e(t_0)=x_0-x_{\mathrm{ref},0},\qquad t\geq t_0,\tag{4.21}$$

$\Delta K(t)=\hat{K}(t)-K$, $K=\left[K_x^{\mathrm{T}},K_r^{\mathrm{T}},\Theta^{\mathrm{T}}\right]^{\mathrm{T}}$, and $(K_x,K_r)\in\mathbb{R}^{n\times m}\times\mathbb{R}^{m\times m}$ verify the matching conditions (4.8) and (4.9).

Theorem 4.4. *Consider the plant's dynamics* (4.6)*, the plant's reference model* (4.7)*, the feedback control law* (4.18)*, and the adaptive laws* (4.14) *and* (4.19)*. If the matching con-*

ditions (4.8) *and* (4.9) *are verified, then the trajectory tracking error* $e(\cdot)$ *and the adaptive gain matrices* $\hat{K}(\cdot)$ *and* $\hat{K}_g(\cdot)$ *are uniformly bounded and* $e(t) \to 0$ *as* $t \to \infty$ *uniformly in* $t_0 \in [0, \infty)$.

Proof: Consider the Lyapunov function candidate

$$V(t, e, \hat{K}, \hat{K}_g) \triangleq e^{\mathrm{T}} P e + \mathrm{tr}\left(\Delta K^{\mathrm{T}} \Gamma^{-1} \Delta K \Lambda\right) + \mathrm{tr}\left(\hat{K}_g^{\mathrm{T}} \Gamma_g^{-1} \hat{K}_g \Lambda\right), \\ (t, e, \hat{K}, \hat{K}_g) \in [t_0, \infty) \times \mathbb{R}^n \times \mathbb{R}^{(n+m+N)\times m} \times \mathbb{R}^{p\times m}, \quad (4.22)$$

where $P \in \mathbb{R}^{n\times n}$ denotes the symmetric, positive-definite solution of (4.12). It follows from (4.12) and (4.21) that

$$\begin{aligned} \dot{V}(t, e, \hat{K}, \hat{K}_g) = -e^{\mathrm{T}} Q e &+ 2\mathrm{tr}\left(\Delta K^{\mathrm{T}} \Gamma^{-1} \dot{\hat{K}}(t) \Lambda\right) \\ &+ 2\mathrm{tr}\left(\Delta K^{\mathrm{T}} \pi(t, x(t), r(t)) e^{\mathrm{T}} P B \Lambda\right) \\ &+ 2\mathrm{tr}\left(\hat{K}_g^{\mathrm{T}} g(t, x(t), e, r(t)) e^{\mathrm{T}} P B \Lambda\right) \\ &+ 2\mathrm{tr}\left(\hat{K}_g^{\mathrm{T}} \Gamma_g^{-1} \dot{\hat{K}}_g(t) \Lambda\right), \end{aligned} \quad (4.23)$$

for all $(t, e, \hat{K}, \hat{K}_g) \in [t_0, \infty) \times \mathbb{R}^n \times \mathbb{R}^{(n+m+N)\times m} \times \mathbb{R}^{p\times m}$, and, applying the adaptive laws (4.14) and (4.19), it holds that

$$\dot{V}(t, e, \hat{K}, \hat{K}_g) = -e^{\mathrm{T}} Q e, \quad (4.24)$$

which is non-positive definite in $(t, e, \hat{K}, \hat{K}_g)$. Therefore, it follows from Corollary 4.1 of [117] that the nonlinear time-varying dynamical system given by (4.21), (4.14), and (4.19) is uniformly Lyapunov stable and hence, $e(\cdot)$, $\hat{K}(\cdot)$, and $\hat{K}_g(\cdot)$ are uniformly bounded.

Next, we apply Barbalat's lemma [117, Lemma 4.1] to prove uniform convergence of the closed-loop trajectory tracking error $e(\cdot)$ to zero. To this goal, we must prove that $\dot{V}(t, e(t), \hat{K}(t), \hat{K}_g(t))$, $t \geq t_0$, is uniformly continuous, and a sufficient condition for the

uniform continuity of $\dot{V}(t, e(t), \hat{K}(t), \hat{K}_g(t))$ is that $\ddot{V}(t, e(t), \hat{K}(t), \hat{K}_g(t))$ is bounded for all $t \geq t_0$ [117, p. 506]. To prove boundedness of the second time derivative of (4.22) along the trajectories of (4.14), (4.19), and (4.21), we note that since A_{ref} is Hurwitz and $r(t)$, $t \geq t_0$, is bounded, it follows from (4.7) that both $x_{\text{ref}}(t)$ and $\dot{x}_{\text{ref}}(t)$ are bounded [117, p. 245]. Hence, $x(t) = e(t) + x_{\text{ref}}(t)$, $t \geq t_0$, is bounded. Furthermore, since $g(\cdot, x, e, r)$ is bounded by assumption for all $t \in [t_0, \infty)$, it follows from (4.18) that $u(\cdot)$ given by (4.20) is bounded and hence, it follows from (4.6) that $\dot{x}(\cdot)$ is bounded. Therefore, $\dot{e}(t) = \dot{x}(t) - \dot{x}_{\text{ref}}(t)$, $t \geq t_0$, is bounded and $\ddot{V}(t, e(t), \hat{K}(t), \hat{K}_g(t)) = -2e^{\mathrm{T}}(t)Q\dot{e}(t)$ is bounded along the trajectories of (4.14), (4.19), and (4.21). Consequently, $\dot{V}(t, e(t), \hat{K}(t), \hat{K}_g(t))$, $t \geq t_0$, is uniformly continuous. Hence, it follows from Barbalat's lemma that $\dot{V}(t, e(t), \hat{K}(t), \hat{K}_g(t)) \to 0$ as $t \to \infty$ and hence, it follows from (4.24) that $e(t) \to 0$ as $t \to \infty$, which concludes the proof. ■

Theorem 4.4 gives the user ample discretion in the control design process by means of the vector function $g(\cdot, \cdot, \cdot, \cdot)$, which can be exploited to meet specifications that could not be enforced directly on the feedback control law by classical MRAC. For instance, setting $p = n$ and $g(t, x, e, r) = K_P e$, $(t, x, e, r) \in [t_0, \infty) \times \mathcal{D} \times \mathbb{R}^n \times \mathbb{R}^m$, where $K_P \in \mathbb{R}^{n \times n}$ is symmetric and positive-definite, introduces a term in (4.18) that is proportional to the trajectory tracking error and hence, allows stronger corrective actions and smaller oscillations of the trajectory tracking error; it follows from (4.14) that in classical MRAC, this effect can be achieved only indirectly by increasing the adaptive rates' norm. The framework proposed in [86], which presents an anti-windup system for the plant (4.6) with $\Lambda = I$ and $\Theta = 0$, can be deduced from Theorem 4.4 by setting $g(t, x, e, r) = \text{sat}(\hat{K}^{\mathrm{T}}(t)\pi(t, x(t), r(t))) - \hat{K}^{\mathrm{T}}(t)\pi(t, x(t), r(t))$, $(t, x, e, r) \in [t_0, \infty) \times \mathcal{D} \times \mathbb{R}^n \times \mathbb{R}^m$, where $\text{sat}(\cdot)$ denotes the *saturation function* [124, p. 19], and relaxing the continuity conditions on $g(\cdot, \cdot, \cdot, \cdot)$.

The continuity conditions on $g(\cdot, \cdot, \cdot, \cdot)$ are sufficient to guarantee the existence of a unique solution of the dynamical system (4.21), (4.14), and (4.19) [124, Th. 3.1], and the boundedness of $g(\cdot, x, e, r)$ for all $t \in [t_0, \infty)$ is needed to guarantee that the solution $x(\cdot)$ of (4.6) with

control input (4.20) does not diverge in time. Comparing the proposed control law (4.18) with the classical MRAC law (4.13), it is apparent that the proposed framework involves an additional set of $p \times m$ ordinary differential equations, namely (4.19). If $g(t, x, e, r) = 0$, $(t, x, e, r) \in [t_0, \infty) \times \mathcal{D} \times \mathbb{R}^n \times \mathbb{R}^m$, then Theorem 4.4 specializes to Theorem 4.3.

4.5 Two-layer model reference adaptive control

4.5.1 Theoretical formulation

In this section, we present a novel adaptive control framework, which we name *two-layer MRAC*. According to this framework, the user introduces an auxiliary reference model in addition to the plant's reference model (4.7). If this auxiliary reference model is designed to converge to the closed-loop system's trajectory tracking error obtained by applying the classical MRAC framework before the transient dynamics of the plant's reference model has decayed, then the two-layer MRAC law steers the closed-loop system's trajectory tracking error to the auxiliary reference model at a rate of convergence that is higher than the rate of convergence of the plant's reference model. Thus, two-layer MRAC addresses one of the drawbacks of MRAC highlighted in Section 4.3, namely imposing arbitrary rates of convergence on the trajectory tracking error, while setting arbitrarily large adaptive rates and choosing the plant's reference model arbitrarily.

Consider the plant (4.6), the plant's reference model (4.7), the vector of feedback variables $\pi(\cdot, \cdot, \cdot)$ given by (4.15), and the *reference model for the trajectory tracking error's transient dynamics*

$$\dot{e}_{\text{ref,transient}}(t) = A_{\text{tran}} e_{\text{ref,transient}}(t), \qquad e_{\text{ref,transient}}(t_0) = x_0 - x_{\text{ref},0}, \qquad t \geq t_0, \tag{4.25}$$

where $A_{\text{tran}} \in \mathbb{R}^{n\times n}$ is Hurwitz. Consider also the *auxiliary reference model*

$$\dot{\varepsilon}(t) = A_{\text{tran}}\varepsilon(t) + B\Lambda\Delta K^{\mathrm{T}}(t)\pi(t, x(t), r(t)) + B\Lambda\Delta K_g^{\mathrm{T}}(t)e(t), \qquad \varepsilon(t_0) = 0, \qquad t \geq t_0, \tag{4.26}$$

where $\Delta K(t) \triangleq \hat{K}(t) - K$, $\Delta K_g(t) \triangleq \hat{K}_g(t) - K_e$, $K = \left[K_x^{\mathrm{T}}, K_r^{\mathrm{T}}, \Theta^{\mathrm{T}}\right]^{\mathrm{T}}$, $(K_x, K_r, K_e) \in \mathbb{R}^{n\times m} \times \mathbb{R}^{m\times m} \times \mathbb{R}^{n\times m}$ verify the matching conditions (4.8), (4.9), and

$$A_{\text{tran}} = A_{\text{ref}} + B\Lambda K_e^{\mathrm{T}}, \tag{4.27}$$

$\hat{K} : [t_0, \infty) \to \mathbb{R}^{(n+m+N)\times m}$ and $\hat{K}_g : [t_0, \infty) \to \mathbb{R}^{n\times m}$ are such that

$$\dot{\hat{K}}(t) = -\Gamma\pi(t, x(t), r(t))\varepsilon^{\mathrm{T}}(t)P_{\text{transient}}B, \qquad \hat{K}(t_0) = \hat{K}_0, \qquad t \geq t_0, \tag{4.28}$$

$$\dot{\hat{K}}_g(t) = -\Gamma_g e(t)\varepsilon^{\mathrm{T}}(t)P_{\text{transient}}B, \qquad \hat{K}_g(t_0) = \hat{K}_{g,0}, \tag{4.29}$$

$\Gamma \in \mathbb{R}^{(n+m+N)\times(n+m+N)}$ and $\Gamma_g \in \mathbb{R}^{n\times n}$ are symmetric and positive-definite, $P_{\text{transient}} \in \mathbb{R}^{n\times n}$ is the symmetric, positive-definite solution of the algebraic Lyapunov equation

$$0 = A_{\text{tran}}^{\mathrm{T}}P_{\text{transient}} + P_{\text{transient}}A_{\text{tran}} + Q_{\text{transient}}, \tag{4.30}$$

and $Q_{\text{transient}} \in \mathbb{R}^{n\times n}$ is symmetric and positive-definite. If the matching conditions (4.8), (4.9), and (4.27) are verified, then the trajectory tracking error $e(\cdot)$ can be computed equivalently either as $e(t) = e_{\text{ref,transient}}(t) + \varepsilon(t)$, $t \geq t_0$, where $e_{\text{ref,transient}}(\cdot)$ verifies (4.25) and $\varepsilon(\cdot)$ verifies (4.26), or as the solution of (4.21) with $p = n$ and $g(t, x, e, r) = e$, $(t, x, e, r) \in [t_0, \infty) \times \mathcal{D} \times \mathbb{R}^n \times \mathbb{R}^m$. The next result, which introduces the two-layer MRAC framework, leverages this equivalence and the analysis of the solution of (4.26) to prove uniform asymptotic convergence of $e(\cdot)$ to zero.

Lemma 4.5. *Consider the reference model for the trajectory tracking error's transient dy-*

namics (4.25)*, the auxiliary reference model* (4.26)*, and the adaptive laws* (4.28) *and* (4.29)*. If the matching conditions* (4.8)*,* (4.9)*, and* (4.27) *are verified, then both the tracking errors* $\varepsilon(\cdot)$ *and* $e(\cdot)$ *and the adaptive gain matrices* $\hat{K}(\cdot)$ *and* $\hat{K}_g(\cdot)$ *are uniformly bounded. Moreover,* $\varepsilon(t) \to 0$ *and* $e(t) \to 0$ *as* $t \to \infty$ *uniformly in* $t_0 \in [0, \infty)$*.*

Proof: Consider the Lyapunov function candidate

$$V(t, \varepsilon, \hat{K}, \hat{K}_g) = \varepsilon^{\mathrm{T}} P_{\mathrm{transient}} \varepsilon + \operatorname{tr}\left(\widetilde{\Delta K}^{\mathrm{T}} \tilde{\Gamma}^{-1} \widetilde{\Delta K} \Lambda\right), \quad (t, \varepsilon, \hat{K}, \hat{K}_g) \in [t_0, \infty) \times \mathbb{R}^n \times \mathbb{R}^{(n+m+N)\times m} \times \mathbb{R}^{n\times m}, \tag{4.31}$$

where $\widetilde{\Delta K}(t) \triangleq [\Delta K(t), \Delta K_g(t)]$ and $\tilde{\Gamma} \triangleq \operatorname{blockdiag}(\Gamma, \Gamma_g)$. Analyzing the time derivative of (4.31) along the trajectories of (4.26), (4.28), and (4.29) and by reasoning as in the proof of Theorem 4.4, it is possible to prove that $\varepsilon(\cdot)$, $\hat{K}(\cdot)$, and $\hat{K}_g(\cdot)$ are uniformly bounded, and $\varepsilon(t) \to 0$ as $t \to \infty$ uniformly in $t_0 \in [0, \infty)$. Moreover, it follows from the triangle inequality [117, Def. 2.6] that

$$\|\varepsilon(t)\| \geq \left| \|e(t)\| - \|e_{\mathrm{ref,transient}}(t)\| \right|, \qquad t \geq t_0, \tag{4.32}$$

and, since $e_{\mathrm{ref,transient}}(\cdot)$ is uniformly bounded and $e_{\mathrm{ref,transient}}(t) \to 0$ as $t \to \infty$, $e(\cdot)$ is uniformly bounded and $e(t) \to 0$ as $t \to \infty$ uniformly in $t_0 \in [0, \infty)$. ■

If $p = n$ and $g(t, x, e, r) = e$, $(t, x, e, r) \in [t_0, \infty) \times \mathcal{D} \times \mathbb{R}^n \times \mathbb{R}^m$, then both Theorem 4.4 and Lemma 4.5 prove the uniform asymptotic convergence of $e(\cdot)$ to zero. However, the adaptive laws (4.14) and (4.19) used in Theorem 4.4 are different than the adaptive laws (4.28) and (4.29) used in Lemma 4.5. Furthermore, Theorem 4.4 proves the asymptotic convergence of $e(\cdot)$ to zero by analyzing the time derivative of a Lyapunov function candidate along the vector field of the closed-loop system's trajectory tracking error, whereas Lemma 4.5 proves the asymptotic convergence of $e(\cdot)$ to zero by analyzing the asymptotic convergence of $\varepsilon(\cdot)$

to zero, that is, the asymptotic convergence of $e(\cdot)$ to $e_{\text{ref,transient}}(\cdot)$, which, in turn, converges to the origin. For having deduced the asymptotic convergence of $e(\cdot)$ to zero indirectly from the analysis of the convergence of $\varepsilon(\cdot)$ to zero, and for having introduced the reference model for the trajectory tracking error's transient dynamics (4.25), the auxiliary reference model (4.26), and the matching condition (4.27), the framework presented in Lemma 4.5 has been named *two-layer MRAC*.

The next result shows how Lemma 4.5 can be applied to set the rate of convergence of the closed-loop system's trajectory tracking error and guarantee that the trajectory tracking error's transient dynamics is faster than the transient dynamics of the plant's reference model.

Lemma 4.6. *Consider the auxiliary reference model (4.26), the reference model for the trajectory tracking error's transient dynamics (4.25), and the adaptive laws (4.28) and (4.29). If (4.8), (4.9), and (4.27) are verified and* $\text{Re}(\lambda_{\max}(A_{\text{tran}})) < \text{Re}(\lambda_{\min}(A_{\text{ref}}))$*, then (4.17) is verified with* $\alpha > -\text{Re}(\lambda_{\max}(A_{\text{ref}}))$.

Proof: Since $e(t) = \varepsilon(t) + e_{\text{ref,tran}}(t)$, $t \geq t_0$, it follows from (4.25), (4.26), the fact that $\log(a+b) = \log a + \log\left(1 + ba^{-1}\right)$, $a, b > 0$, and Proposition 4.2 that

$$
\begin{aligned}
&\log\left(\|e(t)\|\right) \\
&\leq \log\left(\gamma_1 \|x_0 - x_{\text{ref},0}\|\right) + \xi(t)(t - t_0) \\
&+ \log\left(1 - \frac{\gamma_1 \|B\Lambda\|}{\gamma_2 \text{Re}(\lambda_{\max}(A_{\text{tran}}))} \sup_{\tau \in [t_0, \infty)} \left\|\Delta K^{\text{T}}(\tau)\pi(\tau, x(\tau), r(\tau)) + \Delta K_g^{\text{T}}(\tau)\left(\varepsilon(\tau) + e_{\text{ref,tran}}(\tau)\right)\right\| \right. \\
&\qquad \left. \cdot \left[e^{\xi(t)(t-t_0)}\gamma_1 \|x_0 - x_{\text{ref},0}\|\right]^{-1}\right), \qquad t \geq t_0, \qquad (4.33)
\end{aligned}
$$

where $\gamma_1 \geq 1$ and $\gamma_2 \in (0, 1)$. Now, it follows from Lemma 4.5 that $\varepsilon(\cdot)$, $\hat{K}(\cdot)$, and $\hat{K}_g(\cdot)$ are uniformly bounded and, since the reference signal $r(\cdot)$ is uniformly bounded, both the closed-loop plant's trajectory $x(\cdot)$ and the reference trajectory $x_{\text{ref}}(\cdot)$ are uni-

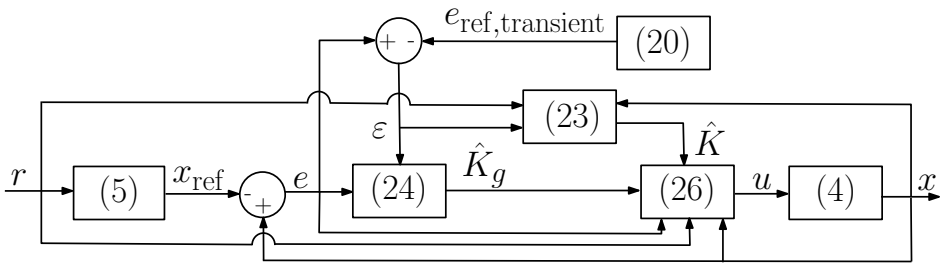

Figure 4.2: Schematic representation of the MRAC architecture outlined in Theorem 4.7. Applying the control law (4.34) and the adaptive laws (4.28) and (4.29), Theorem 4.7 guarantees uniform boundedness of both the closed-loop system's trajectory error $e(\cdot)$ and the adaptive gain matrices $\hat{K}(\cdot)$ and $\hat{K}_g(\cdot)$. Moreover, Theorem 4.7 guarantees uniform asymptotic convergence of $e(\cdot)$ to zero. Lastly, Theorem 4.7 guarantees that $x(\cdot)$ reaches $x_{\text{ref}}(\cdot)$ before the transient dynamics of the plant's reference model (4.7) has decayed.

formly bounded, and $\pi(\cdot, x(\cdot), r(\cdot))$ is uniformly bounded. Furthermore, it follows from Lemma 4.5 that $e(t)$, $t \geq t_0$, is uniformly bounded and $e(t) \to 0$ as $t \to \infty$ uniformly in $t_0 \in [0, \infty)$. Therefore, since $\lim_{t\to\infty} \dfrac{\log\left(1 - ke^{\lambda(t-t_0)}\right)}{t_0 - t} = 0$ for all $k > 0$ and $\lambda < 0$, and $\lim_{t\to\infty} \xi(t) = \text{Re}(\lambda_{\max}(A_{\text{tran}}))$, it follows from (4.33) and Definition 4.1 that $\alpha_{\max}(e(\cdot)) \geq -\text{Re}(\lambda_{\max}(A_{\text{tran}})) > -\text{Re}(\lambda_{\min}(A_{\text{ref}})) \geq -\text{Re}(\lambda_{\max}(A_{\text{ref}})) > 0$, and the result is proven. ■

Lemma 4.6 shows that if $\text{Re}(\lambda_{\max}(A_{\text{tran}})) < \text{Re}(\lambda_{\min}(A_{\text{ref}}))$, then the two-layer MRAC framework introduced by Lemma 4.5 allows to impose the user-defined rate of convergence on the trajectory tracking error $e(\cdot)$ that is faster than the rate of convergence of the reference trajectory $x_{\text{ref}}(\cdot)$. These two results are synthesized by the next theorem, which is the main result of this section. For the statement of this result, it is worthwhile to explicitly present the control law

$$\phi(\pi(t, x, r), e, \hat{K}, \hat{K}_g) = \hat{K}^{\mathrm{T}}\pi(t, x, r) + \hat{K}_g^{\mathrm{T}} e,$$
$$(t, x, e, r, \hat{K}, \hat{K}_g) \in [t_0, \infty) \times \mathcal{D} \times \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^{(n+m+N)\times m} \times \mathbb{R}^{n\times m}, \tag{4.34}$$

that is deduced from (4.18) with $p = n$ and $g(t, x, e, r) = e$, $(t, x, e, r) \in [t_0, \infty) \times \mathcal{D} \times \mathbb{R}^n \times \mathbb{R}^m$.

Theorem 4.7. *Consider the plant's dynamics* (4.6)*, the plant's reference model* (4.7)*, the*

reference model for the trajectory tracking error's transient dynamics (4.25)*, the feedback control law* (4.34)*, and the adaptive laws* (4.28) *and* (4.29)*. If the matching conditions* (4.8)*,* (4.9)*, and* (4.27) *are verified and* $\mathrm{Re}(\lambda_{\max}(A_{\mathrm{tran}})) < \mathrm{Re}(\lambda_{\min}(A_{\mathrm{ref}}))$*, then the trajectory tracking error* $e(\cdot)$ *and the adaptive gain matrices* $\hat{K}(\cdot)$ *and* $\hat{K}_g(\cdot)$ *are uniformly bounded,* $e(t) \to 0$ *as* $t \to \infty$ *uniformly in* $t_0 \in [0, \infty)$*, and* (4.17) *is verified with* $\alpha > -\mathrm{Re}(\lambda_{\min}(A_{\mathrm{ref}}))$*.*

Figure 4.2 provides a schematic representation of the control architecture outlined by Theorem 4.7. In addition to uniform boundedness of the closed-loop system's trajectory tracking error and of the adaptive gains and uniform asymptotic convergence of the trajectory tracking error, which are already guaranteed by the classical MRAC architecture, the control architecture provided by Theorem 4.7 also allows to enforce the user-defined rate of convergence, while setting the adaptive rates arbitrarily high. Remarkably, this result has been achieved without any constraints on the adaptive rates, without modifying the user-defined reference model's dynamics, without employing barrier Lyapunov functions, and without using estimators of the plant's nonlinearities, as required by alternative frameworks such as [82, 83, 84, 91, 92, 98, 99, 100, 101, 102, 103, 105, 106] to name a few.

4.5.2 Illustrative numerical example

In this section, we provide a numerical example to illustrate the applicability of Theorem 4.7. The roll dynamics of an unmanned aerial vehicle is captured by [125, pp. 59-64]

$$\begin{bmatrix} \dot{\varphi}(t) \\ \dot{\rho}(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \theta_1 & \theta_2 \end{bmatrix} \begin{bmatrix} \varphi(t) \\ \rho(t) \end{bmatrix} + \begin{bmatrix} 0 \\ \theta_6 \end{bmatrix} \left(u(t) + \begin{bmatrix} \theta_3 \\ \theta_4 \\ \theta_5 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} |\varphi(t)|\rho(t) \\ |\rho(t)|\rho(t) \\ \varphi^3(t) \end{bmatrix} \right),$$

$$[\varphi(0), \rho(0)]^{\mathrm{T}} = [\varphi_0, \rho_0]^{\mathrm{T}}, \qquad t \geq 0, \quad (4.35)$$

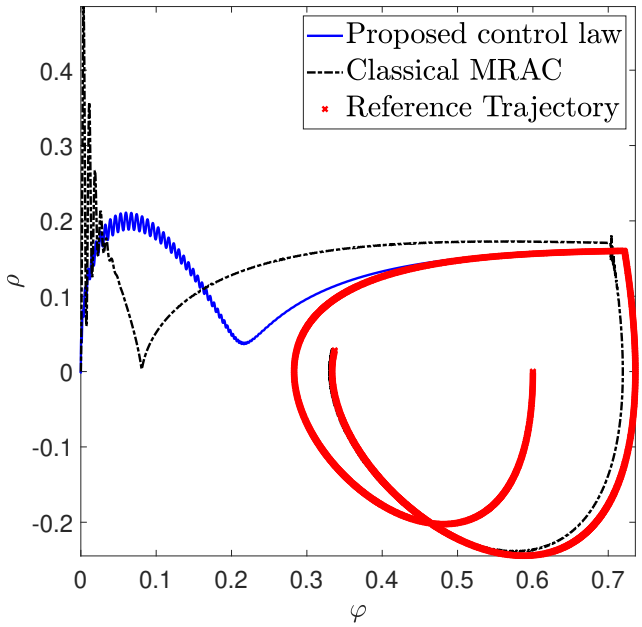

Figure 4.3: Closed-loop plant's trajectory obtained applying the control law (4.18) with $p = n$ and $g(t, x, e, r) = e$, $(t, x, e, r) \in [t_0, \infty) \times \mathcal{D} \times \mathbb{R}^n \times \mathbb{R}^m$, and the adaptive laws (4.28) and (4.29), and closed-loop plant's trajectory obtained applying the classical MRAC law (4.13) with adaptive law (4.14). By applying Theorem 4.7 and setting $\mathrm{Re}(\lambda_{\max}(A_{\mathrm{tran}})) = \sigma \mathrm{Re}(\lambda_{\min}(A_{\mathrm{ref}}))$ with $\sigma = 2$, the closed-loop plant's trajectory reaches the reference trajectory before the closed-loop plant's trajectory obtained applying Theorem 4.3.

where $\varphi(t) \in \mathbb{R}$, $t \geq 0$, denotes the roll angle, $\rho(t) \in \mathbb{R}$ denotes the roll rate, $u(t) \in \mathbb{R}$ denotes the moment needed to deflect the aileron, $\theta_1, \ldots, \theta_5 \in \mathbb{R}$, and $\theta_6 > 0$. In particular, both θ_1 and θ_2 capture the effect of the aerodynamic moments acting on the vehicle, and $[\theta_3, \theta_4, \theta_5][|\varphi|\rho, |\rho|\rho, \varphi^3]^{\mathrm{T}}$ captures undesired aerodynamic moments induced by the aileron deflection; we consider $\theta_1, \ldots, \theta_6$ as unknown. The nonlinear plant (4.35) is in the same form as (4.6) with $n = 2$, $m = 1$, $N = 3$, $\mathcal{D} = \mathbb{R}^n$, $x = [\varphi, \rho]^{\mathrm{T}}$, $A = \begin{bmatrix} 0 & 1 \\ \theta_1 & \theta_2 \end{bmatrix}$, $B = [0, 1]^{\mathrm{T}}$, $\Lambda = \theta_6$, $\Theta = [\theta_3, \theta_4, \theta_5]^{\mathrm{T}}$, $\Phi(t, x) = [|\varphi|\rho, |\rho|\rho, \varphi^3]^{\mathrm{T}}$, $x_0 = [\varphi_0, \rho_0]^{\mathrm{T}}$, and $t_0 = 0$.

Our goal is to design an MRAC law so that the closed-loop trajectory $x(\cdot)$ eventually tracks the trajectory $x_{\mathrm{ref}}(\cdot)$ of the plant's reference model (4.7) with reference command input

$$r(t) = 0.2 \left(\frac{t}{2\pi} - \left\lfloor \frac{t}{2\pi} \right\rfloor \right), \qquad t \geq 0, \tag{4.36}$$

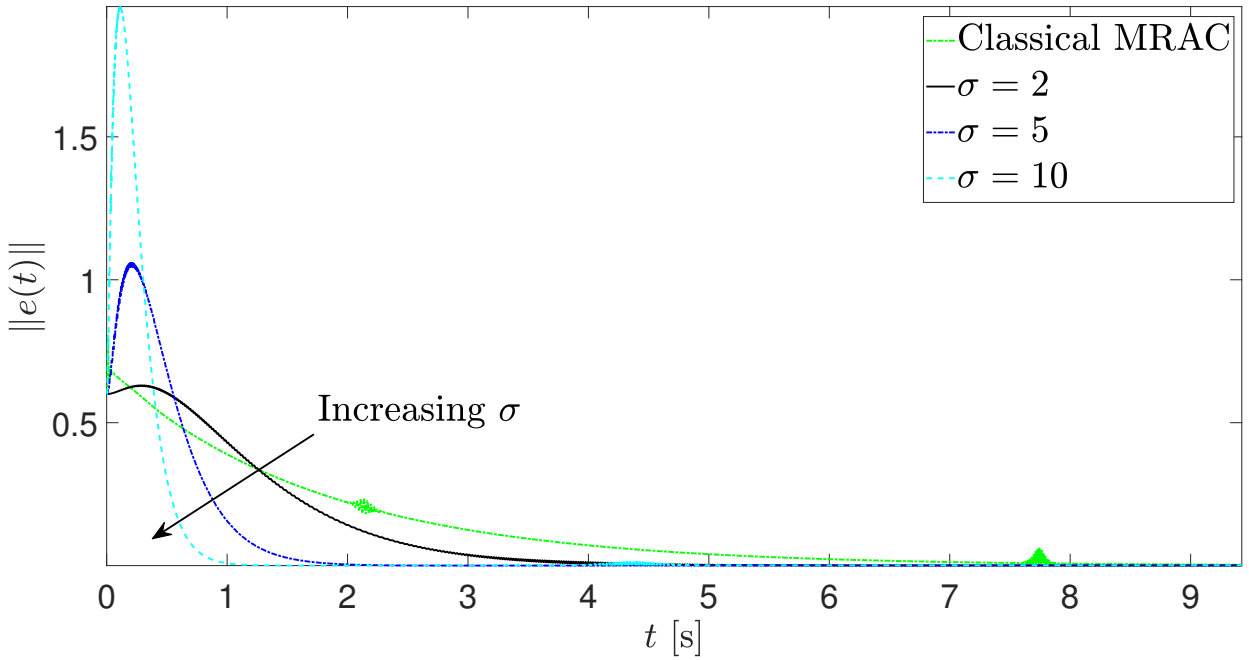

Figure 4.4: Norm of the trajectory tracking error obtained by applying the proposed two-layer MRAC framework, and norm of the trajectory tracking error obtained by applying the classical MRAC framework.

capturing a *sawtooth function*, where $\lfloor t \rfloor \triangleq \max\{m \in \mathbb{Z} : m \leq t\}$ denotes the *floor function*. Furthermore, we want to guarantee that the trajectory tracking error's transient dynamics is faster than the transient dynamics of the plant's reference model. These design specifications can be met by applying Theorem 4.7.

$$\text{Let } A_{\text{ref}} = \begin{bmatrix} 0 & 1 \\ -k_1 & -k_2 \end{bmatrix} B_{\text{ref}} = \begin{bmatrix} 0 \\ k_3 \end{bmatrix}, \text{ and } A_{\text{tran}} = \begin{bmatrix} 0 & 1 \\ -\left[\sigma \text{Re}\left(\lambda_{\min}(A_{\text{ref}})\right)\right]^2 & 2\sigma \text{Re}\left(\lambda_{\min}(A_{\text{ref}})\right) \end{bmatrix},$$

where $k_1, k_2, k_3 > 0$ and $\sigma > 1$. The matrix A_{ref} is Hurwitz, the pair $(A_{\text{ref}}, B_{\text{ref}})$ is controllable, the matching conditions (4.8), (4.9), and (4.27) are verified by $K_x = \left[\frac{\theta_1 - k_1}{\theta_6}, \frac{\theta_2 - k_2}{\theta_6}\right]^{\text{T}}$, $K_r = \dfrac{k_3}{\theta_6}$, and $K_e = \theta_6^{-1}\left[k_1 + k_3, k_2 + k_4\right]^{\text{T}}$, and $\text{Re}(\lambda_{\max}(A_{\text{tran}})) = \sigma \text{Re}(\lambda_{\min}(A_{\text{ref}})) < 0$. Therefore, the hypotheses of Theorem 4.7 are verified.

Let $\theta_1 = -0.018$, $\theta_2 = 0.015$, $\theta_3 = -0.062$, $\theta_4 = 0.009$, $\theta_5 = 0.021$, $\theta_6 = 0.75$, $k_1 = 1.0002$, $k_2 = 1.7218$, $k_3 = 5$, $x_0 = 0$, $x_{\text{ref},0} = [0.6, 0]^{\text{T}}$, $\Gamma = 5 \cdot 10^5 I_6$, $\Gamma_g = 5 \cdot 10^5 I_2$, $Q = I_2$, and $Q_{\text{transient}} = Q$. Figure 4.3 shows the solution $x(t)$, $t \in [0, 3\pi]$ s, of (4.35) obtained applying

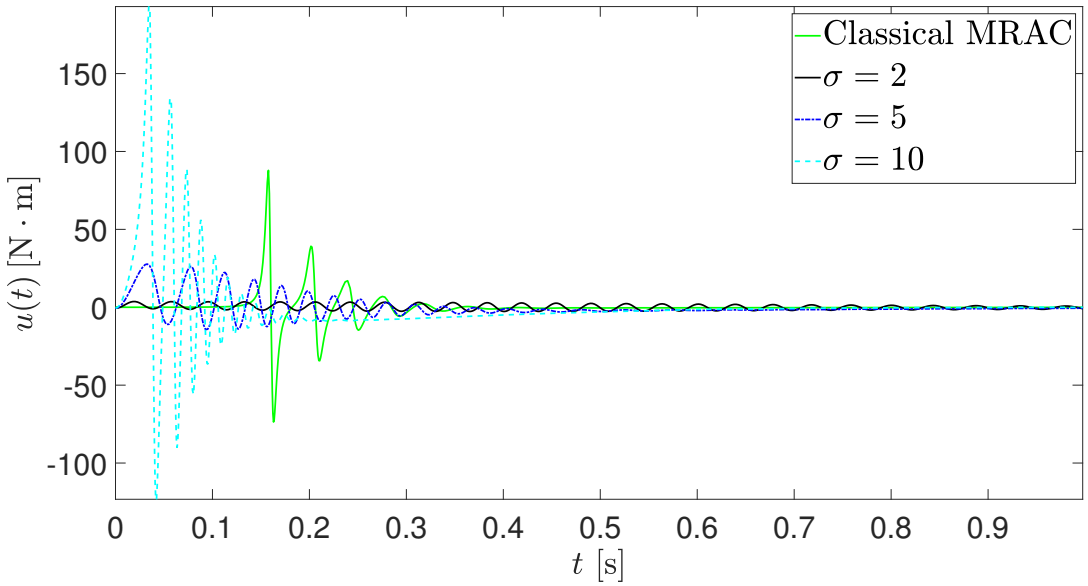

Figure 4.5: Control inputs obtained by applying Theorem 4.7 with $\text{Re}(\lambda_{\max}(A_{\text{tran}})) = \sigma\text{Re}(\lambda_{\min}(A_{\text{ref}}))$ and $\sigma \in \{2, 5, 10\}$, and Theorem 4.3. Larger values of σ imply larger control inputs.

the feedback control law (4.34) and the adaptive laws (4.28) and (4.29) with $\sigma = 2$, the solution $x(t)$ of (4.35) obtained applying the control law (4.13) and the classical adaptive law (4.14), and the solution $x_{\text{ref}}(t)$ of (4.7). Figure 4.4 shows the norm of the trajectory tracking error applying the proposed two-layer model referenced adaptive control framework with $\sigma \in \{2, 5, 10\}$ and the norm of the trajectory tracking error applying the classical MRAC framework. It is apparent that applying the proposed approach, the trajectory tracking error converges to zero faster than applying the classical MRAC framework. Moreover, larger values of σ guarantee faster convergence of the trajectory tracking error to zero, but also imply larger overshoots. Lastly, Figure 4.5 shows the control inputs obtained applying the proposed model referenced adaptive control law with $\sigma \in \{2, 5, 10\}$ and applying the classical MRAC framework. Larger values of σ imply larger control inputs and, applying the proposed adaptive control framework with $\sigma < 7.8095$, the absolute value of the maximum control input required by the proposed framework is smaller than the absolute value of the maximum control input required by the classical MRAC framework.

4.6 Model reference adaptive control laws in the presence of constraints

4.6.1 Theoretical formulation

In this section, we design MRAC laws for prescribed performance. These adaptive laws guarantee both uniform asymptotic convergence of the closed-loop system's trajectory tracking error and uniform boundedness of the trajectory tracking error and of the adaptive rates, exploit barrier Lyapunov functions to enforce user-defined constraints on both the closed-loop system's trajectory tracking error and control input, and leverage the proposed two-layer MRAC framework to impose a user-defined rate of convergence on the trajectory tracking error. To present these MRAC laws, consider the plant (4.6) with $\Lambda = I_m$, the plant's reference model (4.7), and the *constraint sets*

$$\mathcal{E} \triangleq \left\{ e \in \mathbb{R}^n : h_e(e^{\mathrm{T}} M e) \geq 0 \right\}, \tag{4.37}$$

$$\mathcal{U} \triangleq \{ u \in \mathbb{R}^m : h_u(u) \geq 0 \}, \tag{4.38}$$

where $M \in \mathbb{R}^{n \times n}$ is a symmetric and positive-definite solution of the inequality

$$A_{\mathrm{ref}}^{\mathrm{T}} M + M A_{\mathrm{ref}} \leq 0, \tag{4.39}$$

$h_e : \mathbb{R} \to \mathbb{R}$, $h_u : \mathbb{R}^m \to \mathbb{R}$, $h_e(0) > 0$, $h_u(0) > 0$, $h_e(\cdot)$ is continuously differentiable for all $e \in \mathcal{E}$, and $h_u(\cdot)$ is continuously differentiable, bounded on compact subsets of $\mathbb{R}^m$, and such that $\frac{\partial h_u(u)}{\partial u}$ is bounded over compact subsets of $\mathbb{R}^m$. For example, if

$$h_e(e^{\mathrm{T}} M e) = \eta_e - e^{\mathrm{T}} e, \qquad e \in \mathbb{R}^n, \tag{4.40}$$

$$h_u(u) = \eta_u - u^{\mathrm{T}} u, \quad u \in \mathbb{R}^m, \tag{4.41}$$

where $\eta_e, \eta_u > 0$, then (4.37) captures a closed ball of radius η_e and (4.38) captures saturation constraints on the control input.

Assuming that $\Lambda = I_m$ in (4.6) is without loss of generality. Indeed, let $\phi(t)$, $t \geq t_0$, denote, for brevity, the feedback control law (4.18) along the trajectories of the closed-loop system, that is, $\phi(\pi(t, x(t), r(t)), g(t, x(t), e(t), r(t)), \hat{K}(t), \hat{K}_g(t))$. Then, $B\Lambda \left[\phi(t) + \Theta^{\mathrm{T}}\Phi(t, x(t))\right] = B\left[\phi(t) + \tilde{\Theta}^{\mathrm{T}}\tilde{\Phi}(t, x(t))\right]$, $(t, x) \in [t_0, \infty) \times \mathcal{D}$, where $\tilde{\Theta}^{\mathrm{T}} \triangleq \left[\Theta^{\mathrm{T}}, (I_m - \Lambda)\right]$ and $\tilde{\Phi}(t, x) \triangleq \left[\Phi^{\mathrm{T}}(t, x), \phi^{\mathrm{T}}(t)\right]^{\mathrm{T}}$. Therefore, assuming that $\Lambda = I_m$ in (4.6) and capturing the plant's parametric and matched uncertainties by $\tilde{\Theta}^{\mathrm{T}}\tilde{\Phi}(t, x)$, $(t, x) \in [t_0, \infty) \times \mathcal{D}$, is equivalent to capturing the plant dynamics by means of (4.6). This equivalence, however, is proven by augmenting the regressor vector and hence, yields at the expense of a larger number of adaptive laws to integrate and a higher computational cost.

For the statement of the results of this section, define $h_e' : \mathbb{R} \to \mathbb{R}$ such that

$$h_e'(e^{\mathrm{T}}Me) \triangleq \left.\frac{\partial h_e(\beta)}{\partial \beta}\right|_{\beta = e^{\mathrm{T}}Me}, \qquad e \in \mathbb{R}^n, \tag{4.42}$$

let

$$V_e(e) \triangleq h_e^{-1}(e^{\mathrm{T}}Me)e^{\mathrm{T}}Pe, \qquad e \in \mathring{\mathcal{E}}, \tag{4.43}$$

$$V_g(t, \hat{K}_g) \triangleq h_u^{-1}(\phi(t))\,\mathrm{tr}^{\frac{1}{2}}\left(\hat{K}_g^{\mathrm{T}}\Gamma_g^{-1}\hat{K}_g\right), \qquad (t, \hat{K}_g) \in [t_0, \infty) \times \mathbb{R}^{p \times m}, \tag{4.44}$$

and consider the adaptive laws

$$\dot{\hat{K}}(t) = -\Gamma\frac{\pi(t, x(t), r(t))}{h_e(e^{\mathrm{T}}(t)Me(t))}e^{\mathrm{T}}(t)\left[P - V_e(e(t))h_e'(e^{\mathrm{T}}(t)Me(t))M\right]B, \ \hat{K}(t_0) = \hat{K}_0, \ t \geq t_0, \tag{4.45}$$

$$\begin{aligned}\dot{\hat{K}}_g(t) = &-\frac{2h_u^2(\phi(t))V_g(t, \hat{K}_g(t))}{h_e(e^{\mathrm{T}}(t)Me(t))}\Gamma_g g(t, x(t), e(t), r(t))e^{\mathrm{T}}(t)\left[P - V_e(e(t))h_e'(e^{\mathrm{T}}(t)Me(t))M\right]B \\ &+ \frac{\gamma^{\mathrm{T}}(t)h_{\mathrm{du}}(t) + \chi}{h_u(\phi(t))}\hat{K}_g(t), \qquad \hat{K}_g(t_0) = \hat{K}_{g,0},\end{aligned} \tag{4.46}$$

where $\pi(\cdot,\cdot,\cdot)$ is given by (4.15), $e(t) \in \mathring{\mathcal{E}}$ denotes the solution of (4.21), $g : [t_0, \infty) \times \mathcal{D} \times \mathcal{E} \times \mathbb{R}^m \to \mathbb{R}^p$ is jointly continuous in its arguments, $g(t, \cdot, \cdot, r)$ is locally Lipschitz continuous in (x, e) uniformly in (t, r) on compact subsets of $[t_0, \infty) \times \mathbb{R}^m$, $g(\cdot, x, e, r)$ is bounded for all $t \in [t_0, \infty)$, both $\Gamma \in \mathbb{R}^{n \times n}$ and $\Gamma_g \in \mathbb{R}^{p \times p}$ are symmetric and positive-definite, $\phi(t) \in \mathring{\mathcal{U}}$, $\hat{K}_g(t_0) \neq 0$, $\gamma(\cdot)$, $h_{\mathrm{du}}(\cdot)$, and χ are such that

$$\dot{h}_u(\phi(t)) = \gamma^{\mathrm{T}}(t) h_{\mathrm{du}}(t) + \beta(t), \tag{4.47}$$

$\gamma : [t_0, \infty) \to \mathbb{R}^{N_{\mathrm{f}}}$ is known, $h_{\mathrm{du}} : [t_0, \infty) \to \mathbb{R}^{N_{\mathrm{f}}}$, $\beta(t) \in [-\beta_{\max}, \beta_{\max}]$ is unknown, $\beta_{\max} \geq 0$, $\gamma^{\mathrm{T}}(t) h_{\mathrm{du}}(t)$ is continuous over $[t_0, \infty)$, and $\chi < -\beta_{\max}$. Given the user-defined parameter $N_{\mathrm{f}} \in \mathbb{N}$, the term $\gamma^{\mathrm{T}}(t) h_{\mathrm{du}}(t)$, $t \geq t_0$, in (4.47) approximates the time derivative of $h_u(\phi(t))$ and can be computed, for instance, using differentiators [126, pp. 235-236]. Several filters, such as the finite impulse response filter [127, Ch. 7] or the fixed-point smoother [128, Ch. 5], can be employed to design differentiators, which guarantee that the *approximation error* $\beta(\cdot)$ in (4.47) is bounded.

Lastly, to state the results of this section, the following two assumptions are made. To formulate these assumptions, recall that the *null-controllable region of* (4.10) *associated to the constraint set* $\mathcal{U}$ is the set $\mathcal{N}_{\mathcal{U}} \subseteq \mathbb{R}^n$ such that if $e(t_0) \in \mathcal{N}_{\mathcal{U}}$, then there exist both $T > t_0$ and $u : [t_0, T] \to \mathbb{R}^m$ such that $u(t) \in \mathcal{U}$, $t \in [t_0, T]$, and $e(T) = 0$ [129]. Furthermore, recall that if K_x, K_r, and Θ were known, then the control input given by (4.11) would guarantee asymptotic convergence of the closed-loop trajectory tracking error.

Assumption 4.6.1. Consider the open-loop trajectory tracking error dynamics (4.10) and the constraint sets $\mathcal{E}$ and $\mathcal{U}$ given by (4.37) and (4.38), respectively. It holds that $\mathcal{E} \subseteq \mathcal{N}_{\mathcal{U}}$.

Assumption 4.6.2. The control input $u(\cdot)$ given by (4.11) is such that $u(t) \in \mathring{\mathcal{U}}$, $t \geq t_0$, and the corresponding trajectory tracking error $e(\cdot)$, which verifies (4.16) with $\Delta K(t) = 0$, $t \geq t_0$, is such that $e(t) \in \mathring{\mathcal{E}}$.

The next theorem proves that the MRAC law (4.18) and the adaptive laws (4.45) and (4.46) enforce the user-defined constraints on both the closed-loop system's trajectory tracking error and the control input given by (4.37) and (4.38), respectively, and guarantee uniform asymptotic convergence of the closed-loop system's trajectory tracking error.

Theorem 4.8. *Consider the plant's dynamics* (4.6) *with* $\Lambda = I_m$*, the plant's reference model* (4.7)*, the constraint sets* (4.37) *and* (4.38)*, the feedback control law* (4.18)*, and the adaptive laws* (4.45) *and* (4.46)*. If* $(e(t_0), \phi(t_0)) \in \mathring{\mathcal{E}} \times \mathring{\mathcal{U}}$*,* $h'_e(e^{\mathrm{T}}Me) \leq 0$*,* $e \in \mathring{\mathcal{E}}$*, Assumptions 4.6.1 and 4.6.2 are verified, and the matching conditions* (4.8) *and* (4.9) *are verified, then both the trajectory tracking error* $e(\cdot)$ *and the adaptive gain matrices* $\hat{K}(\cdot)$ *and* $\hat{K}_g(\cdot)$ *are uniformly bounded,* $(e(t), \phi(t)) \in \mathring{\mathcal{E}} \times \mathring{\mathcal{U}}$*,* $t \geq t_0$*, and* $e(t) \to 0$ *as* $t \to \infty$ *uniformly in* $t_0 \in [0, \infty)$*.*

Proof of Theorem 4.8: The proof of this result is divided in two parts. Firstly, we assume that $(e(t), \phi(t)) \in \mathring{\mathcal{E}} \times \mathring{\mathcal{U}}$, $t \geq t_0$, and proceed as in the proof of Theorem 4.4 to prove both the boundedness of $e(\cdot)$, $\hat{K}(\cdot)$, and $\hat{K}_g(\cdot)$, and the asymptotic convergence of $e(\cdot)$ to zero. Then, we use a contradiction argument to prove that if $(e(t_0), \phi(t_0)) \in \mathring{\mathcal{E}} \times \mathring{\mathcal{U}}$, then $(e(t), \phi(t)) \in \mathring{\mathcal{E}} \times \mathring{\mathcal{U}}$, $t \geq t_0$.

Let Q be symmetric, positive-definite, and such that $Q \geq \zeta I_n$, where $\zeta > 0$. Define also

$$Q_1(e) \triangleq V_e(e) h'_e(e^{\mathrm{T}}Me) \left(A_{\mathrm{ref}}^{\mathrm{T}} M + M A_{\mathrm{ref}}\right), \qquad e \in \mathring{\mathcal{E}}, \tag{4.48}$$

$$\tilde{Q}(e) \triangleq Q + Q_1(e), \tag{4.49}$$

where $V_e(\cdot)$ is given by (4.43). It follows from (4.39) that $A_{\mathrm{ref}}^{\mathrm{T}} M + M A_{\mathrm{ref}}$ is symmetric and nonpositive-definite and hence, since $h'_e(e^{\mathrm{T}}Me) \leq 0$, $e \in \mathring{\mathcal{E}}$, $Q_1(\cdot)$ is symmetric and nonnegative-definite. Therefore, $\tilde{Q}(\cdot)$ is symmetric, $\tilde{Q}(e) \geq \zeta I_n$, $e \in \mathring{\mathcal{E}}$, and it follows from (4.48), (4.49), and (4.12) that

$$-\tilde{Q}(e) = A_{\mathrm{ref}}^{\mathrm{T}} \left[P - V_e(e) h'_e(e^{\mathrm{T}}Me) M\right] + \left[P - V_e(e) h'_e(e^{\mathrm{T}}Me) M\right] A_{\mathrm{ref}}, \qquad e \in \mathring{\mathcal{E}}. \tag{4.50}$$

Next, assume that $(e(t), \phi(t)) \in \mathring{\mathcal{E}} \times \mathring{\mathcal{U}}$, $t \geq t_0$, and consider the barrier Lyapunov function candidate

$$V(t, e, \hat{K}, \hat{K}_g) = V_e(e) + \operatorname{tr}\left(\Delta K^{\mathrm{T}} \Gamma^{-1} \Delta K\right) + V_g(t, \hat{K}_g), \tag{4.51}$$

for all $(t, e, \hat{K}, \hat{K}_g) \in [t_0, \infty) \times \mathring{\mathcal{E}} \times \mathbb{R}^{(n+m+N)\times m} \times \mathbb{R}^{p\times m}$. Since $z_1^{\mathrm{T}} z_2 = \operatorname{tr}(z_2 z_1^{\mathrm{T}})$ for all $z_1, z_2 \in \mathbb{R}^n$, it follows from (4.12), (4.21), (4.50), (4.47), (4.45), and (4.46) that

$$\begin{aligned}
\dot{V}(t, e, \hat{K}, \hat{K}_g) = &-h_e^{-1}(e^{\mathrm{T}} Q e) e^{\mathrm{T}} \tilde{Q}(e) e \\
&+ 2h_e^{-1}(e^{\mathrm{T}} M e) \operatorname{tr}\Big(\Delta K^{\mathrm{T}} \pi(t, x(t), r(t)) e^{\mathrm{T}} [P - V_e(e) h_e'(e^{\mathrm{T}} M e) M] B\Big) \\
&+ 2\operatorname{tr}\Big(\Delta K^{\mathrm{T}} \Gamma^{-1} \dot{\hat{K}}(t)\Big) \\
&+ 2h_e^{-1}(e^{\mathrm{T}} M e) \operatorname{tr}\Big(\hat{K}_g^{\mathrm{T}}(t) g(t, x(t), e, r(t)) e^{\mathrm{T}} [P - V_e(e) h_e'(e^{\mathrm{T}} M e) M] B\Big) \\
&- h_u^{-2}(\phi(t)) \left[\gamma^{\mathrm{T}}(t) h_{\mathrm{du}}(t) + \beta(t)\right] \operatorname{tr}^{\frac{1}{2}}\left(\hat{K}_g^{\mathrm{T}} \Gamma_g^{-1} \hat{K}_g\right) \\
&+ h_u^{-1}(\phi(t)) \operatorname{tr}^{-\frac{1}{2}}\left(\hat{K}_g^{\mathrm{T}} \Gamma_g^{-1} \hat{K}_g\right) \operatorname{tr}\left(\hat{K}_g^{\mathrm{T}} \Gamma_g^{-1} \dot{\hat{K}}_g(t)\right) \\
= &-h_e^{-1}(e^{\mathrm{T}} Q e) e^{\mathrm{T}} \tilde{Q}(e) e + \frac{\chi - \beta(t)}{h_u^2(\phi(t))} \operatorname{tr}^{\frac{1}{2}}\left(\hat{K}_g^{T} \Gamma_g^{-1} \hat{K}_g\right), \\
&\qquad (t, e, \hat{K}, \hat{K}_g) \in [t_0, \infty) \times \mathring{\mathcal{E}} \times \mathbb{R}^{(n+m+N)\times m} \times \mathbb{R}^{p\times m}.
\end{aligned} \tag{4.52}$$

Therefore, $\dot{V}(\cdot, \cdot, \cdot, \cdot)$ is non-positive definite and, by proceeding as in the proof of Theorem 4.4, we verify that if $(e(t), \phi(t)) \in \mathring{\mathcal{E}} \times \mathring{\mathcal{U}}$, $t \geq t_0$, then $e(\cdot)$, $\hat{K}(\cdot)$, and $\hat{K}_g(\cdot)$ are uniformly bounded, and $e(t) \to 0$ as $t \to \infty$ uniformly in $t_0 \in [0, \infty)$.

Next, we prove that if $e(t_0) \in \mathring{\mathcal{E}}$ and $u(t) \in \mathring{\mathcal{U}}$, $t \geq t_0$, with $u(\cdot)$ given by (4.20), then $e(t) \in \mathring{\mathcal{E}}$. To this goal, assume *ad absurdum* that there exists a finite-time $T_1 > t_0$ such that $h_e(e^{\mathrm{T}}(T_1) M e(T_1)) = 0$, that is, such that $e(T_1) \in \partial\mathcal{E}$. Since $0 \in \mathring{\mathcal{E}}$ and P is symmetric and positive-definite, it holds that $\lim_{t \to T_1^-} e^{\mathrm{T}}(t) P e(t) = e^{\mathrm{T}}(T_1) P e(T_1) > 0$. Therefore, it follows from (4.43) that $V_e(e(t)) \to \infty$ as $t \to T_1^-$ and, since $V(\cdot, \cdot, \cdot, \cdot)$ is the sum of positive-

definite terms, $V(t,e(t),\hat{K}(t),\hat{K}_g(t)) \to \infty$ as $t \to T_1^-$. However, it follows from (4.52) that $V(t,e(t),\hat{K}_x(t),\hat{K}_g(t))$, $t \geq t_0$, is uniformly bounded along the trajectories of (4.21), (4.45), and (4.46), which is a contradiction. Thus, $e(t) \in \mathring{\mathcal{E}}$, $t \geq t_0$.

Next, we prove that if $\phi(t_0) \in \mathring{\mathcal{U}}$ and $e(t) \in \mathring{\mathcal{E}}$, $t \geq t_0$, then $\phi(t) \in \mathring{\mathcal{U}}$. To this goal, assume *ad absurdum* that there exists a finite-time $T_2 > t_0$ such that $h_u(\phi(T_2)) = 0$, that is, such that $\phi(T_2) \in \partial\mathcal{U}$. Now, two alternative cases must be considered, namely $\operatorname{tr}\left(\hat{K}_g^{\mathrm{T}}(T_2)\Gamma_g^{-1}\hat{K}_g(T_2)\right) \neq 0$ and $\operatorname{tr}\left(\hat{K}_g^{\mathrm{T}}(T_2)\Gamma_g^{-1}\hat{K}_g(T_2)\right) = 0$. If $\lim_{t\to T_2^-} h_u(\phi(t)) = 0$ and $\lim_{t\to T_2^-} \operatorname{tr}\left(\hat{K}_g^{\mathrm{T}}(t)\Gamma_g^{-1}\hat{K}_g(t)\right) \neq 0$, then it follows from (4.44) that $V_g(t,\hat{K}_g(t)) \to \infty$ as $t \to T_2^-$, which implies that $V(t,e(t),\hat{K}(t),\hat{K}_g(t)) \to \infty$ as $t \to T_2^-$. However, this conclusion contradicts the uniform boundedness of $V(t,e(t),\hat{K}(t),\hat{K}_g(t))$, $t \geq t_0$, and hence $\phi(t) \in \mathring{\mathcal{U}}$. Alternatively, if $h_u(\phi(T_2)) = 0$ and $\operatorname{tr}\left(\hat{K}_g^{\mathrm{T}}(T_2)\Gamma_g^{-1}\hat{K}_g(T_2)\right) = 0$, then it follows from l'Hôpital's rule that

$$\lim_{t\to T_2^-} V_g(t,\hat{K}_g(t)) = \lim_{t\to T_2^-} \frac{\frac{\mathrm{d}}{\mathrm{d}t}\operatorname{tr}^{\frac{1}{2}}\left(\hat{K}_g^{\mathrm{T}}(t)\Gamma_g^{-1}\hat{K}_g(t)\right)}{\dot{h}_u(\phi(t))}, \tag{4.53}$$

and it follows from (4.46) that

$$\lim_{t\to T_2^-} \frac{\mathrm{d}}{\mathrm{d}t}\operatorname{tr}^{\frac{1}{2}}\left(\hat{K}_g^{\mathrm{T}}(t)\Gamma_g^{-1}\hat{K}_g(t)\right) = \lim_{t\to T_2^-} \left[\left(\gamma^{\mathrm{T}}(t)h_{\mathrm{du}}(t) + \chi\right) V_g(t,\hat{K}_g(t))\right], \tag{4.54}$$

$$\lim_{t\to T_2^-} \dot{h}_u(\phi(t)) = \omega(T_2) + \lim_{t\to T_2^-} \left[\frac{\gamma^{\mathrm{T}}(t)h_{\mathrm{du}}(t) + \chi}{h_u(\phi(t))} \cdot \frac{\partial h_u(\phi(t))}{\partial u}\hat{K}_g^{\mathrm{T}}(t)g(t,x(t),e(t),r(t))\right], \tag{4.55}$$

where $\omega(t) \triangleq \frac{\partial h_u(\phi(t))}{\partial u}\left(\dot{\hat{K}}^{\mathrm{T}}(t)\pi(t,x(t),r(t)) + \hat{K}^{\mathrm{T}}(t)\dot{\pi}(t,x(t),r(t))\right)$, $t \geq t_0$. Now, two instances must be considered, namely $\omega(T_2) \neq \gamma^{\mathrm{T}}(T_2)h_{\mathrm{du}}(T_2) + \chi$ and $\omega(T_2) = \gamma^{\mathrm{T}}(T_2)h_{\mathrm{du}}(T_2) + \chi$. Assume that $\omega(T_2) \neq \gamma^{\mathrm{T}}(T_2)h_{\mathrm{du}}(T_2) + \chi$. In this case, since $\gamma^{\mathrm{T}}(t)h_{\mathrm{du}}(t)$ is continuous over $t \in [t_0, T_2]$, it follows from the Weierstrass' theorem [117, Th. 2.11] that $\gamma^{\mathrm{T}}(t)h_{\mathrm{du}}(t)$

is bounded over $[t_0, T_2]$. Thus, it follows from (4.53)–(4.55) that $\lim_{t\to T_2^-} V_g^{-1}(t, \hat{K}_g(t)) = \left[1 - \frac{\omega(T_2)}{\gamma^{\mathrm{T}}(T_2)h_{\mathrm{du}}(T_2)+\chi}\right]^{-1} \lim_{t\to T_2^-} \frac{\partial h_u(\phi(t))}{\partial u} \mathrm{sign}(\hat{K}_g^{\mathrm{T}}(t)) g(t, x(t), e(t), r(t)) = 0$, where the *signum function of* $\hat{K}_g$ is defined so that if $\hat{K}_g \neq 0$, then $\mathrm{sign}(\hat{K}_g) = \hat{K}_g \|\hat{K}_g\|^{-1}_{\mathrm{F},\Gamma_g^{-1}}$, and if $\hat{K}_g = 0$, then $\mathrm{sign}(\hat{K}_g) = 0$, and $\|\hat{K}_g\|_{\mathrm{F},\Gamma_g^{-1}} \triangleq \mathrm{tr}^{\frac{1}{2}}(\hat{K}_g^{\mathrm{T}}\Gamma_g^{-1}\hat{K}_g)$ denotes the *weighted Frobenius norm of* $\hat{K}_g$. Therefore, if $\omega(T_2) \neq \gamma^{\mathrm{T}}(T_2)h_{\mathrm{du}}(T_2) + \chi$, then $\lim_{t\to T_2^-} V_g(t, \hat{K}_g(t)) = \infty$, which contradicts the uniform boundedness of $V(t, e(t), \hat{K}_x(t), \hat{K}_g(t))$, $t \geq t_0$, along the trajectories of (4.21), (4.45), and (4.46). Alternatively, if $\omega(T_2) = \gamma^{\mathrm{T}}(T_2)h_{\mathrm{du}}(T_2) + \chi$, then it follows from (4.53) and (4.54) that $\lim_{t\to T_2^-} \dot{h}_u(\phi(t)) = \lim_{t\to T_2^-} \gamma^{\mathrm{T}}(t)h_{\mathrm{du}}(t) + \chi$. However, in this case, it follows from (4.47) that $\lim_{t\to T_2^-} \beta(t) = \chi$, which contradicts the assumption that χ is such that $\chi < -\beta_{\max}$. Therefore, $\phi(t) \in \mathring{\mathcal{U}}$, $t \geq t_0$.

Using similar arguments, the proof is completed by showing that if $(e(t_0), \phi(t_0)) \in \mathring{\mathcal{E}} \times \mathring{\mathcal{U}}$ and Assumptions 4.6.1 and 4.6.2 hold, then $(e(t), \phi(t)) \in \mathring{\mathcal{E}} \times \mathring{\mathcal{U}}$, $t \geq t_0$. ■

It is worthwhile to note that if $h_u(u) = 1$, $u \in \mathbb{R}^m$, then $\mathcal{U} = \mathbb{R}^m$ and Assumptions 4.6.1 and 4.6.2 are always verified. In this case, by setting $h_{\mathrm{du}}(t) = 0$, $t \geq t_0$, and $\chi = 0$, Theorem 4.8 allows to enforce the constraints on the trajectory tracking error. We also note that if no constraint is imposed, that is, if $h_e(e^{\mathrm{T}}Me) = 1$, $e \in \mathbb{R}^n$, and $h_u(u) = 1$, $u \in \mathbb{R}^m$, then (4.45) and (4.46) reduce to (4.14) and (4.19), respectively, and Theorem 4.8 reduces to Theorem 4.4. Lastly, we note that the term $\dfrac{h_u^2(\phi(t))}{h_e(e^{\mathrm{T}}(t)Me(t))}$, $t \geq t_0$, in (4.46) captures the effect of competing design objectives namely, constraining the control input by means of $h_u(\cdot)$, which induces smaller values of $\|\dot{\hat{K}}_g(\cdot)\|$, and constraining the trajectory tracking error by means of $h_e(\cdot)$, which induces larger values of $\|\dot{\hat{K}}_g(\cdot)\|$.

Assumption 4.6.1 postulates the existence of a control input that meets the user-defined constraints on both the trajectory tracking error and the control input, and Assumption 4.6.2 postulates that the control input given by (4.11) meets the user-defined constraints. If the uncontrolled trajectory tracking error dynamics (4.10) is unstable, then there may not exist a control input that regulates the trajectory tracking error and meets the constraints captured

by $\mathcal{U}$ in (4.38) at all times [130, pp. 222-223]. Assumptions 4.6.1 and 4.6.2 guarantee that there exist control inputs, including (4.11), that meet the user-defined constraints. To verify Assumption 4.6.1, the null-controllable region of (4.10) may be characterized by applying some of the techniques presented in [129, 131, 132, 133], which leverage the analysis of the basin of attraction of Lyapunov functions and the analysis of time-optimal trajectories for the plant dynamics. The assumption that $h_e(0) > 0$ implies that $0 \in \mathring{\mathcal{E}}$, and for the problems addressed in this work, the origin is always contained in the null-controllable region of (4.10). Indeed, the pair (A, B) is controllable and hence, it follows from the matching condition (5.17) with $\Lambda = I_m$ that the pair (A_{ref}, B) is controllable, and it follows from Theorem 6.1 of [134] that the null-controllable region $\mathcal{N}_{\mathcal{U}}$ of (4.10) associated to $\mathcal{U}$ is an open, connected set containing the origin.

Theorem 4.8 does not require to specify the function $g(\cdot, \cdot, \cdot, \cdot)$ in the control law (4.18) and in the adaptive law (4.46). Therefore, $g(\cdot, \cdot, \cdot, \cdot)$ may serve as a design parameter to enforce additional design objectives. The next theorem is the main result of this chapter and shows how the two-layer MRAC framework introduced in Section 4.5 can be used to create MRAC laws for prescribed performance, that is, MRAC laws that guarantee boundedness of the adaptive gains, constrain both the trajectory tracking error and the control input, enforce uniform asymptotic convergence to zero of the trajectory tracking error, and assure that the trajectory tracking error's rate of convergence is arbitrarily high. For the statement of this result, consider the constraint sets (4.37) and (4.38) and the adaptive laws

$$\dot{\hat{K}}(t) = -\Gamma \frac{\pi(t, x(t), r(t))}{h_e(e^{\mathrm{T}}(t) M e(t))} \varepsilon^{\mathrm{T}}(t) \Big[P_{\text{transient}} - \tilde{V}_e(e(t), \varepsilon(t)) h_e'(e^{\mathrm{T}}(t) M e(t)) M \Big] B,$$

$$\hat{K}(t_0) = \hat{K}_0, \qquad t \geq t_0, \tag{4.56}$$

$$\dot{\hat{K}}_g(t) = -\frac{2 h_u^2(\phi(t)) \tilde{V}_g(t, \hat{K}_g(t))}{h_e(e^{\mathrm{T}}(t) M e(t))} \Gamma_g e(t) \varepsilon^{\mathrm{T}}(t) \Big[P_{\text{transient}} - \tilde{V}_e(e(t), \varepsilon(t)) h_e'(e^{\mathrm{T}}(t) M e(t)) M \Big] B$$
$$+ \frac{\gamma^{\mathrm{T}}(t) h_{\text{du}}(t) + \chi}{2 h_u(\phi(t))} \hat{K}_g(t), \qquad \hat{K}_g(t_0) = \hat{K}_{g,0}, \tag{4.57}$$

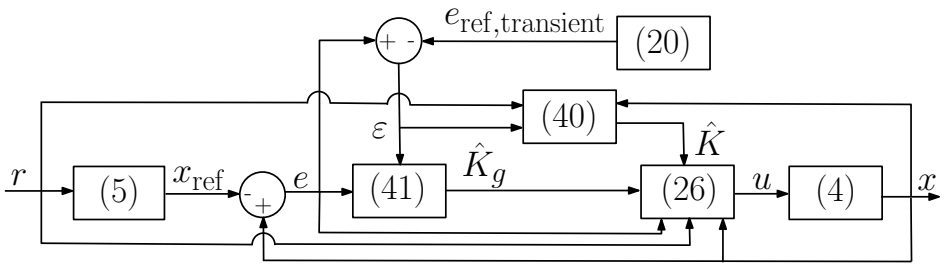

Figure 4.6: Schematic representation of the MRAC architecture outlined in Theorem 4.9. Applying the control law (4.34) and the adaptive laws (4.56) and (4.57), Theorem 4.9 guarantees uniform boundedness of the adaptive gain matrices $\hat{K}(\cdot)$ and $\hat{K}_g(\cdot)$. Moreover, Theorem 4.9 guarantees uniform asymptotic convergence of $e(\cdot)$ to zero. Additionally, Theorem 4.9 guarantees that $x(\cdot)$ reaches $x_{\text{ref}}(\cdot)$ before the transient dynamics of the plant's reference model has decayed. Lastly, Theorem 4.9 guarantees that the user-defined constraints on the closed-loop system's trajectory tracking error and the control input captured by (4.37) and (4.38), respectively.

where both $\Gamma \in \mathbb{R}^{(n+m+N)\times(n+m+N)}$ and $\Gamma_g \in \mathbb{R}^{n\times n}$ are symmetric and positive-definite, $\varepsilon(\cdot)$ denotes the solution of (4.26) with $\Lambda = I_n$, $e(t) \in \mathring{\mathcal{E}}$ denotes the solution of (4.21) with $p = n$ and $g(t,x,e,r) = e$, $(t,x,e,r) \in [t_0,\infty) \times \mathcal{D} \times \mathring{\mathcal{E}} \times \mathbb{R}^m$, $P_{\text{transient}} \in \mathbb{R}^{n\times n}$ denotes the symmetric, positive-definite solution of (4.30), $\phi(t) \in \mathring{\mathcal{U}}$ denotes, for brevity, the feedback control law (4.34) along the trajectories of the closed-loop system, that is, $\phi(\pi(t,x(t),r(t)),e(t),\hat{K}(t),\hat{K}_g(t))$,

$$\tilde{V}_e(e,\varepsilon) \triangleq h_e^{-1}(e^{\mathrm{T}}Me)\varepsilon^{\mathrm{T}}P_{\text{transient}}\varepsilon, \qquad (e,\varepsilon) \in \mathring{\mathcal{E}} \times \mathbb{R}^n, \tag{4.58}$$

$$\tilde{V}_g(t,\hat{K}_g) \triangleq h_u^{-1}(\phi(t))\,\mathrm{tr}^{\frac{1}{2}}\left(\Delta K_g^{\mathrm{T}}\Gamma_g^{-1}\Delta K_g\right), \qquad (t,\hat{K}_g) \in [t_0,\infty) \times \mathbb{R}^{n\times m}, \tag{4.59}$$

$\Delta K_g(t) = \hat{K}_g(t) - K_e$, K_e verifies the matching condition (4.27) with $\Lambda = I_n$, $\hat{K}_g(t_0) \neq K_e$, $\gamma(\cdot)$ and $h_{\text{du}}(\cdot)$ are such that (4.47) is verified, and $\chi < -\beta_{\max}$. Lastly, we enunciate the following assumption.

Assumption 4.6.3. The trajectory $e_{\text{ref,tran}}(\cdot)$ of the reference model for the trajectory tracking error's transient dynamics (4.25) is such that $e_{\text{ref,tran}}(t) \in \mathring{\mathcal{E}}$, $t \geq t_0$.

Theorem 4.9. *Consider the plant's dynamics* (4.6) *with* $\Lambda = I_m$*, the plant's reference model* (4.7)*, the reference model for the trajectory tracking error's transient dynamics* (4.25)*, the*

constraint sets (4.37) *and* (4.38)*, the feedback control law* (4.34)*, and the adaptive laws* (4.56) *and* (4.57)*. If* $(e(t_0), \phi(t_0)) \in \mathring{\mathcal{E}} \times \mathring{\mathcal{U}}$*,* $h_e'(e^{\mathrm{T}}Me) \leq 0$*,* $e \in \mathring{\mathcal{E}}$*, Assumptions 4.6.1–4.6.3 are verified, the matching conditions* (4.8)*,* (4.9)*, and* (4.27) *are verified with* $\Lambda = I_m$*, and* $\mathrm{Re}(\lambda_{\max}(A_{\mathrm{tran}})) < \mathrm{Re}(\lambda_{\min}(A_{\mathrm{ref}}))$*, then both the trajectory tracking error* $e(\cdot)$ *and the adaptive gain matrices* $\hat{K}(\cdot)$ *and* $\hat{K}_g(\cdot)$ *are uniformly bounded. Moreover,* $(e(t), \phi(t)) \in \mathring{\mathcal{E}} \times \mathring{\mathcal{U}}$*,* $t \geq t_0$*, and* $e(t) \to 0$ *and* $\varepsilon(t) \to 0$ *as* $t \to \infty$ *uniformly in* $t_0 \in [0, \infty)$*. Lastly,* (4.17) *is verified with* $\alpha > -\mathrm{Re}(\lambda_{\max}(A_{\mathrm{ref}}))$*.*

Theorem 4.9 allows to impose both the rate of convergence on the closed-loop plant's trajectory and user-defined bounds on the closed-loop system's trajectory tracking error and the control input, and guarantees both uniform boundedness of the adaptive gains and uniform asymptotic convergence of the trajectory tracking error to zero. To the authors' best knowledge, there is no result within the MRAC framework that allows to meet all these specifications concurrently. The proof of Theorem 4.9 follows the same mechanisms as the proofs of Lemma 4.5 and Theorems 4.7 and 4.8 and hence, its proof is omitted for brevity. Figure 4.6 provides a schematic representation of the control architecture outlined by Theorem 4.9. Remarkably, the only difference between the control scheme presented in Figure 4.2 and the control scheme presented in Figure 4.6 lies in the adaptive laws employed to impose the multiple, possibly competing, user-defined requirements on the closed-loop plant's trajectory, the control input, and the trajectory tracking error.

4.6.2 Illustrative numerical example

In the following, we revisit the problem of controlling the roll dynamics of an unmanned aerial vehicle discussed in Section 4.5.2, and illustrate the applicability of Theorem 4.9.

Specifically, consider the plant

$$\begin{bmatrix} \dot{\varphi}(t) \\ \dot{\rho}(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \theta_1 & \theta_2 \end{bmatrix} \begin{bmatrix} \varphi(t) \\ \rho(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 0.75 \end{bmatrix} \left(u(t) + \begin{bmatrix} \theta_3 \\ \theta_4 \\ \theta_5 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} |\varphi(t)|\rho(t) \\ |\rho(t)|\rho(t) \\ \varphi^3(t) \end{bmatrix} \right) + \xi(t),$$

$$[\varphi(0), \rho(0)]^{\mathrm{T}} = [\varphi_0, \rho_0]^{\mathrm{T}}, \qquad t \geq 0, \quad (4.60)$$

and note that if $\xi(t) = 0$, $t \geq 0$, then (4.60) is in the same form as (4.6) with $n = 2$, $m = 1$, $N = 3$, $\mathcal{D} = \mathbb{R}^n$, $x = [\varphi, p]^{\mathrm{T}}$, $A = \begin{bmatrix} 0 & 1 \\ \theta_1 & \theta_2 \end{bmatrix}$, $B = [0, 0.75]^{\mathrm{T}}$, $\Lambda = 1$, $\Theta = [\theta_3, \theta_4, \theta_5]^{\mathrm{T}}$, $\Phi(t, x) = [|\varphi|\rho, |\rho|\rho, \varphi^3]^{\mathrm{T}}$, $x_0 = [\varphi_0, p_0]^{\mathrm{T}}$, and $t_0 = 0$. The unmatched uncertainty $\xi : [0, \infty) \to \mathbb{R}^2$ captures a Gaussian white noise characterized by a frequency of 100 Hz and $\mathcal{L}_\infty$-norm of 0.05.

Our goal is to design an MRAC law so that the closed-loop trajectory $x(\cdot)$ eventually tracks the trajectory $x_{\mathrm{ref}}(\cdot)$ of the plant's reference model (4.7) with reference command input (4.36). Furthermore, we wish to enforce that $(e(t), u(t)) \in \mathring{\mathcal{E}} \times \mathring{\mathcal{U}}$, $t \geq t_0$, where $\mathcal{E}$ and $\mathcal{U}$ are given by (4.37) and (4.38), respectively, with $h_e(\cdot)$ and $h_u(\cdot)$ given by (4.40) and (4.41), respectively, so that $\|e(t)\| \in [0, \sqrt{\eta_e})$ and $u(t) \in (-\sqrt{\eta_u}, \sqrt{\eta_u})$. Lastly, we want to guarantee that the trajectory tracking error's transient dynamics is faster than the transient dynamics of the plant's reference model. These design specifications can be met by applying Theorem 4.9 with $A_{\mathrm{ref}} = \begin{bmatrix} 0 & 1 \\ -k_1 & -k_2 \end{bmatrix}$, $A_{\mathrm{tran}} = \begin{bmatrix} 0 & 1 \\ k_4 & k_5 \end{bmatrix}$, $\theta_1 = -0.018$, $\theta_2 = 0.015$, $\theta_3 = -0.062$, $\theta_4 = 0.009$, $\theta_5 = 0.021$, $k_1 = 1.0002$, $k_2 = 1.7218$, $k_4 \triangleq -\left[\sigma \mathrm{Re}\left(\lambda_{\min}(A_{\mathrm{ref}})\right)\right]^2$, $k_5 \triangleq 2\sigma \mathrm{Re}\left(\lambda_{\min}(A_{\mathrm{ref}})\right)$, $\sigma = 2$, $x_0 = 0$, $x_{\mathrm{ref},0} = [0.6, 0]^{\mathrm{T}}$, $\Gamma = 5 \cdot 10^5 I_6$, $\Gamma_g = 5 \cdot 10^5 I_2$, $Q = I_2$, and $Q_{\mathrm{transient}} = Q$. Furthermore, let $k = 50$, $\eta_e = 0.85$, $\eta_u = 9$, $\beta_{\max} = 1$, and $\chi = -1$.

The stochastic unmatched uncertainty $\xi(\cdot)$ has been introduced in the plant model (4.60) to validate the proposed theoretical results also in realistic applications, wherein external,

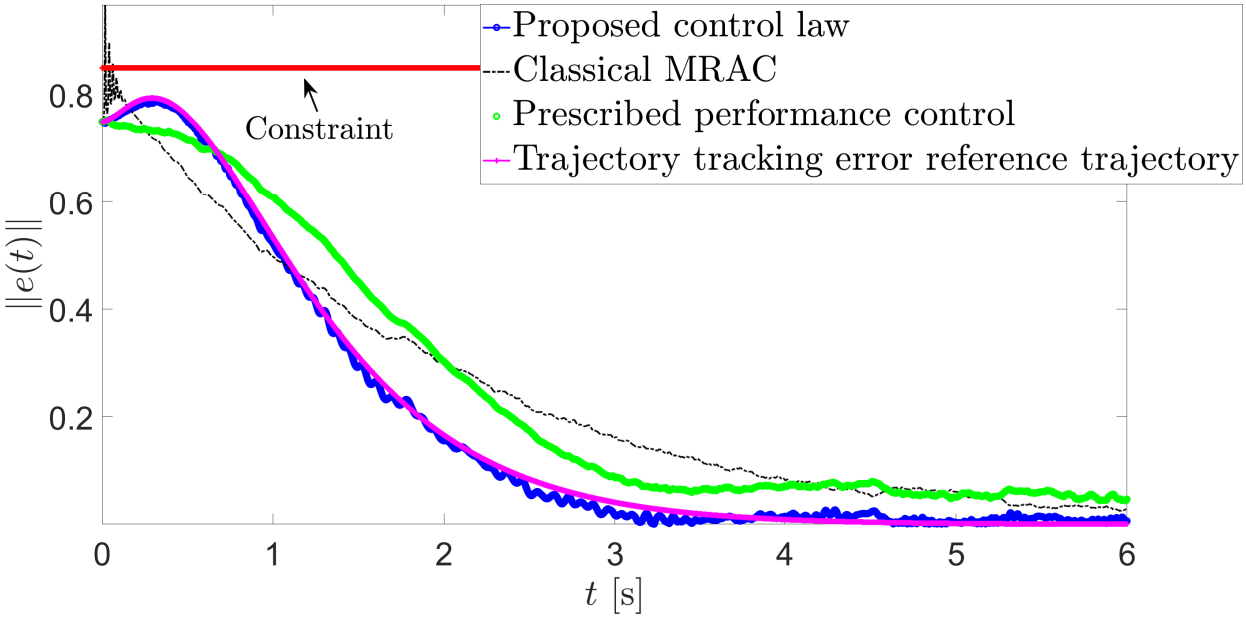

Figure 4.7: Norms of the trajectory tracking errors obtained by applying Theorems 4.9, 4.3, and the prescribed performance control method. Applying either the proposed adaptive control framework or the prescribed performance control framework, the norm of the trajectory tracking error does not violate the user-defined constraint. Applying the classical MRAC framework, the constraint on the trajectory tracking error is violated. Applying Theorem 4.9, the trajectory tracking error converges to zero faster than applying prescribed performance control, which in turn is faster than the trajectory tracking error achieved by applying Theorem 4.3. Remarkably, applying Theorem 4.9, the plot of the norm of the trajectory tracking error $e(\cdot)$ substantially overlaps with the plot of the norm of the trajectory tracking error reference trajectory $e_{\mathrm{ref,tran}}(\cdot)$.

non-deterministic disturbances are unavoidable. To increase the realism of the proposed simulations and further challenge Theorem 4.9, the state vector $x(\cdot)$ employed in the feedback control law (4.18) and the adaptive laws (4.56) and (4.57) is corrupted by an additive disturbance, namely a Gaussian white noise characterized by a frequency of 100 Hz and $\mathcal{L}_\infty$-norm of 0.02.

To further validate the results of the proposed control law, we compare the proposed control law to the prescribed performance control law presented in [135], which guarantees *a priori* user-defined transient performance without modifying the reference model or the reference signal, despite uncertainties in the plant parameters. Applying the prescribed

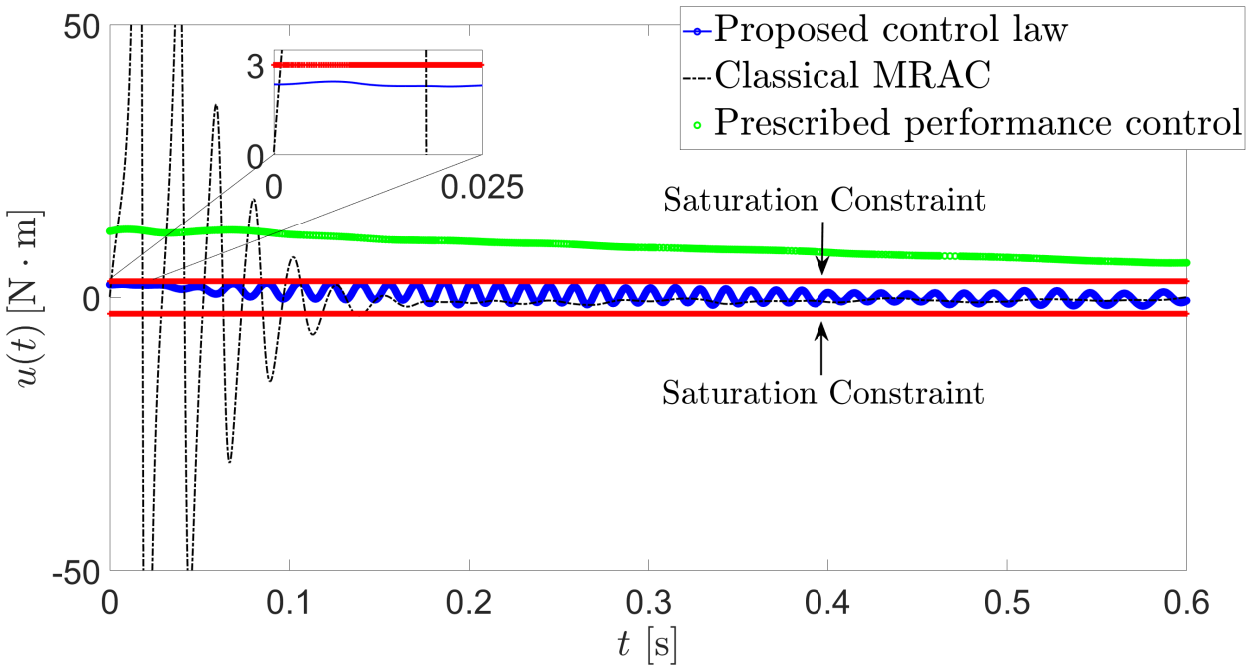

Figure 4.8: Control inputs obtained applying Theorems 4.9, 4.3, and the prescribed performance control method. Applying the proposed adaptive control framework, the control input verifies the saturation constraints. These constraints are violated applying both the classical MRAC law and the prescribed performance control technique.

performance control framework, the control input is given by

$$u(t) = -\frac{k_{\text{ppc}} z_{\text{ppc}}(t)}{s_{\text{ppc}}(t)} - \frac{\hat{\Theta}_{\text{ppc}}(t)}{2\eta_{\text{ppc}}^2}\Phi^{\mathrm{T}}(x(t))\Phi(x(t)) - \frac{\hat{\epsilon}_{\text{ppc}}^2(t) s_{\text{ppc}}(t)}{\hat{\epsilon}_{\text{ppc}}(t)|s_{\text{ppc}}(t)| + \sigma_{1,\text{ppc}}}, \qquad t \geq 0, \quad (4.61)$$

where $k_{\text{ppc}}, \eta_{\text{ppc}}, \sigma_{1,\text{ppc}} > 0$ are user-defined,

$$z_{\text{ppc}}(t) \triangleq \Lambda_{\text{ppc}} \ln\left(1 + \frac{e_{\text{ppc}}(t)}{1 - e_{\text{ppc}}(t)}\right) + s_{\text{ppc}}(t)\left[[0,1]\left(x(t) - x_{\text{ref}}(t)\right) - e_{\text{ppc}}(t)\dot{\rho}(t)\right], \quad (4.62)$$

$$s_{\text{ppc}}(t) \triangleq \frac{1}{2\rho(t)}\left[\frac{1}{e_{\text{ppc}}(t) + 1} - \frac{1}{e_{\text{ppc}}(t) - 1}\right], \quad (4.63)$$

$\Lambda_{\text{ppc}} > 0$ is user-defined, $e_{\text{ppc}}(t) \triangleq \rho^{-1}(t)\,[1,0]\,[x(t) - x_{\text{ref}}(t)]$, $\rho(t) \triangleq (\rho_0 - \rho_\infty)e^{-\lambda_{\text{ppc}}(t-t_0)} + \rho_\infty$ captures denotes user-defined constraints on the trajectory tracking error, $\rho_0, \rho_\infty > 0$, $\lambda_{\text{ppc}} >$

0 captures the user-defined decay rate,

$$\dot{\hat{\Theta}}_{\rm ppc}(t) = \Gamma_{1,\rm ppc} s_{\rm ppc}(t) \left(\frac{z_{\rm ppc}^2(t)}{2\eta_{\rm ppc}^2} \Phi^{\rm T}(x(t)) \Phi(x(t)) - \sigma_{2,\rm ppc} \hat{\Theta}_{\rm ppc}(t) \right), \qquad \hat{\Theta}_{\rm ppc}(0) = \hat{\Theta}_{\rm ppc,0}, \tag{4.64}$$

$$\dot{\hat{\epsilon}}_{\rm ppc}(t) = \Gamma_{2,\rm ppc} s_{\rm ppc}(t) \left[|z_{\rm ppc}(t)| - \sigma_{3,\rm ppc} \hat{\epsilon}_{\rm ppc}(t) \right], \qquad \hat{\epsilon}_{\rm ppc}(0) = \hat{\epsilon}_{\rm ppc,0}, \tag{4.65}$$

and $\Gamma_{1,\rm ppc}, \Gamma_{2,\rm ppc}, \sigma_{2,\rm ppc}, \sigma_{3,\rm ppc} > 0$ are user-defined. For additional details, see [87, 88, 135]. To ensure that the system converges to the reference model before the transient has decayed, the decay rate of the constraint function is set as $\lambda_{\rm ppc} = -\mathrm{Re}(\lambda_{\min}(A_{\rm ref}))$, and the additional user-defined parameters are $k_{\rm ppc} = 1$, $\eta_{\rm ppc} = 0.01$, $\sigma_{\rm ppc} = 0.1$, $\Lambda_{\rm ppc} = 1$, $\Gamma_{\rm ppc1} = \Gamma_{\rm ppc2} = 1000$, $\rho_0 = 1$, and $\rho_\infty = 0.03$.

Figure 4.7 shows the norm of the trajectory tracking error applying the Theorem 4.3, that is, (4.13) with adaptive law (4.14), the MRAC law for prescribed performance proposed in Theorem 4.9, that is, (4.34) with adaptive laws (4.56) and (4.57), and prescribed performance control, that is, (4.61) with adaptive laws (4.64) and (4.65). Employing either Theorem 4.9 or the prescribed performance control technique, the trajectory tracking error does not violate the constraint captured by (4.37) and (4.40), whereas applying Theorem 4.3, the constraint on the trajectory tracking error is violated. Applying Theorem 4.9, the trajectory tracking error converges to zero faster than applying either Theorem 4.3 or using the prescribed performance control method, and the prescribed performance control guarantees a higher convergence rate than the classical MRAC law. It is worthwhile to note that the plot of the norm of the trajectory tracking error $e(\cdot)$ substantially overlaps with the plot of the norm of the trajectory tracking error reference trajectory $e_{\rm ref,tran}(\cdot)$.

Figure 4.8 shows the control input applying the proposed framework, the classical MRAC law, and the prescribed performance control law. It is apparent that, applying Theorem 4.9, the control input verifies the saturation constraints captured by (4.38) and (4.41) at

all time. These constraints, however, are exceeded by both the classical MRAC law and the prescribed performance control law. Applying (4.34) with adaptive laws (4.56) and (4.57) and setting $\lambda_{\text{ppc}} = -\lambda_{\min}(A_{\text{tran}})$, even faster convergence can be achieved. However, simulation results show that, in this case, the trajectory tracking error violates the user-defined constraints due to large excursions of the error velocity, and even larger control inputs are realized. The differentiator employed in this example to approximate the time derivative of $h_u(\phi(t)) = \eta_u^2 - \phi^2(t)$, $t \in [0, \infty)$, is based on the classical Parks-McClellan optimal finite impulse response filter [127, Ch. 7].

In this chapter, we presented a unified MRAC framework, which guarantees that both the trajectory tracking error and the adaptive gains are uniformly bounded, the trajectory tracking error asymptotically converges to zero, the plant's actuators do not saturate, the plant's trajectory does not exceed its null-controllable region, and the plant's trajectory tracking error reaches the reference trajectory at the user-defined rate of convergence. The proposed framework is readily applicable to UAVs performing payload deliveries, where their limited actuation presents the need for explicit control saturation constraints, flying close to buildings and persons creates a need for a priori trajectory tracking error guarantees to ensure safety, and unknown, and guaranteed transient performance ensures safe flights during adaptation to new and potentially unknown payloads. The next chapter is focused on the next generation of UAV, where in addition to unknowns in the system, the UAV may have to perform contact-based interactions with surfaces upon the delivery of a payload.

Chapter 5

Model reference adaptive control of switched dynamical systems with applications to aerial robotics

In Chapter 3, a UAV routing scenario was considered, where UAVs were tasked with performing some deliveries of payloads. In the future, UAVs may be tasked with going one step further and performing contact-based tasks such as installing the payload. Even robust adaptive control techniques such as those presented in Chapter 4 may be unable to ensure stability in the presence of discontinuous dynamics which can be caused by contacting a hard surface. This chapter presents an MRAC law for unknown nonlinear switched plants that must follow the trajectory of user-defined linear switched reference models. The effectiveness of the proposed control architecture is proven in two mathematical frameworks, that is, analyzing Carathéodory and Filippov solutions of discontinuous differential equations. The proposed control law is validated numerically on the roll dynamics of a reconfigurable delta-wing aircraft where the discontinuities occur with increasing frequency. The proposed method is experimentally validated by tasking an aerial robot, a quadrotor with tilting propellers, with mounting a camera of unknown inertial properties onto a vertical surface.

5.1 Literature review

The dynamics of numerous mechanical and electronic systems is subject to instantaneous changes and are best captured by switched dynamical systems, that is, differential equations with discontinuous right-hand sides. Examples of switched dynamical models involve mechanical systems subject to velocity jumps and force discontinuities [136, 137]. Furthermore, the closed-loop dynamics of systems regulated by discontinuous control algorithms such as time-optimal control laws [138, pp. 110-117], variable structure control laws [124, pp. 552-579], [139, 140, 141], and supervisory control architectures [142, 143, 144, 145] may be captured by switched models. Substantial complexity in the analysis and control synthesis of discontinuous dynamical systems is given by the fact that their solutions may not exist or may not be unique [124, Ch. 3]. Furthermore, the notion of solution of a switched differential equation is not univocal. Indeed, Carathéodory [146], Filippov [147], Krasovskii [148], and Euler [149] solutions, to name a few, have been introduced to better capture the behavior of different classes of discontinuous dynamical systems; for additional details, see [150, 151, 152, 153] and the references therein.

In this chapter, an adaptive control law is designed for unknown nonlinear switched plants so that their trajectories track the trajectories of user-defined switched reference models. The mapping between the switching signal and the plant dynamics is considered as unknown. The switching signal is assumed to be a known function of time since in model reference adaptive control, the reference model is user-defined and independent of the plant state.

The effectiveness of the proposed model reference adaptive control law is proven in two alternative frameworks, that is, by considering Carathéodory and Filippov solutions of discontinuous differential equations. These two frameworks have been chosen since both Carathéodory and Filippov solutions of discontinuous dynamical systems are usually more suitable to analyze the dynamics of mechanical systems. Indeed, both Carathéodory and Filippov solutions are absolutely continuous and hence, have bounded variations over bounded

time intervals, and their time derivatives exist almost everywhere [154, pp. 127-130]. Applying the Carathéodory framework, we prove that if the switching signal is characterized by an arbitrarily small, but non-zero, dwell-time, then solutions of both the trajectory tracking error's and the adaptive gains' dynamics exist, are unique, and are defined almost everywhere over the semi-infinite time horizon, and the plant trajectory asymptotically converges to the reference model's trajectory. Employing the Filippov framework, we prove that if the switching signal is Lebesgue integrable and has countably many points of discontinuity, then maximal solutions of both the trajectory tracking error and the adaptive gains dynamics exist and are defined almost everywhere on the semi-infinite time horizon, and the trajectory tracking error converges to zero asymptotically. Considering Filippov solutions, the switching signal may have zero dwell-time, but the uniqueness of the solutions of the trajectory tracking error's and the adaptive gains' dynamics cannot be proven.

The theoretical framework employed in this chapter does not allow to control plants whose dimensions vary as arbitrary functions of time. However, the proposed framework allows to address those problems, wherein the plant's dynamics is in partial-state equilibrium [117, Def. 4.1] for some values of the switching signal. In these cases, the dynamics of those components of the state vector that are at equilibrium can be disregarded, and only the dynamics of those components of the state vector that are not at equilibrium is regulated by applying the proposed framework.

To the authors' best knowledge, this is the first work to deduce model reference adaptive control laws for unknown nonlinear switched plants and switched reference models employing the Carathéodory and the Filippov frameworks. Furthermore, the proposed model reference adaptive control laws are unique for their ability to regulate unknown nonlinear plants without any restrictions on the dwell-time. The design of model reference controls for linear switched systems was addressed in [155, 156]. The authors in [157] proposed an H_∞-based adaptive control law for switched linear dynamical systems. Supervisory control architectures involving multiple adaptive control laws have been proposed in

[158, 159, 160, 161, 162, 163, 164, 165] to regulate linear uncertain dynamical systems, while guaranteeing user-defined levels of performance in the transient regime. In [166, 167, 168], the authors devised an adaptive sliding mode control law for switched dynamical systems that are linear in the parameters and subject to external disturbances, and proved the validity of their results in the Filippov and the Krasovskii frameworks. The design of model reference adaptive control laws for uncertain nonlinear plants was presented in [169] employing the average dwell-time method under asynchronous switching, and [170] analyzing classical solutions of the closed-loop system. An adaptive controller for nonlinear systems, which does not rely on any restrictions on the dwell-time, has been presented in [171]. However, this result is achieved by assuming that the linear portion of the plant dynamics is the same for all switched systems. An adaptive control law for nonlinear switched systems with arbitrary switching is presented in [172]. However, these results apply if there exists a diffeomorphism such that the plant dynamics is equivalent to a cascaded dynamical system.

The effectiveness of the proposed results is firstly verified numerically. Specifically, we present a numerical example that involves the design of a control law for the roll dynamics of a delta-wing aircraft that can switch between two alternative configurations, namely a stable and less responsive configuration and an unstable and more responsive configuration. Reconfigurable aircraft are particularly advantageous for those applications, wherein the vehicle must operate in multiple flight regimes by rapidly changing its geometric and aerodynamic properties [173, 174, 175, 176, 177]. However, rapid or instantaneous changes in the aircraft configuration or the reference model underlying the control architecture may induce instabilities. The robustness of the proposed model reference adaptive control law is challenged by assuming that the aircraft aerodynamic coefficients are unknown in all configurations and by switching arbitrarily fast both the plant's and the reference model's dynamics. Control algorithms for morphing-wing aircraft have been investigated in [178] using an H_∞ control framework, [179] employing a backstepping approach, and [180] using a variable structure switched control law. Furthermore, a supervisory control architecture has been presented

in [180] to regulate a vertical take-off and landing aircraft modeled as switched dynamical systems. None of the control techniques for uncertain, switched, nonlinear plants that we surveyed is suitable to regulate plants in the same form as the dynamical model in the proposed numerical example. Thus, the performance of the proposed model reference adaptive control law is compared to the performance of the classical model reference adaptive control law [75, Ch. 9] obtained considering only one of the two aircraft configurations. In particular, it is shown how, considering only the dynamical model for the less responsive configuration, the classical model reference adaptive control law is unable to regulate the plant dynamics, and the trajectory tracking error diverges. Alternatively, considering only the dynamical model for the more responsive configuration, the trajectory tracking error, the control effort, and the computational time are considerably larger than the trajectory tracking error, the control effort, and the computational time achieved by applying the proposed adaptive law.

The effectiveness of the proposed results is verified also by means of flight tests. These flight tests involve an autonomous aerial robot, that is, a tilt-rotor quadcopter equipped with a robotic arm, whose task is to install a camera of unknown mass on a vertical surface. The aerial robot holds the camera by means of a suction cup, and linear strip fasteners are used to attach the camera to the vertical surface. A switched dynamical model is employed to capture the aerial robot's dynamics. Indeed, as soon as the robotic arm impacts the vertical surface, the vehicle's forward motion is impeded by reaction forces. Furthermore, while the camera is being installed on the vertical surface, the aerial robot's yaw and roll dynamics and lateral motion are constrained by the suction cup and the linear fabric strip fasteners. Flight tests results clearly show that the proposed model reference adaptive control law guarantees successful completion of the assigned task despite uncertainties on the aerial manipulator's dynamics. The problem of designing control algorithms for aerial systems interacting with hard surfaces, such as walls and floors, has been investigated recently by applying feedback-linearizing control laws within a hybrid systems framework [181, 182, 183]. It is worthwhile to remark that, imposing some conditions on the minimum dwell-time, the

results in [181, 182, 183] allow instantaneous increases of the trajectory tracking error at switching times, whereas the proposed adaptive control framework does not restrict the plant's minimum dwell time and does not allow instantaneous variations in the trajectory tracking error. To the authors' best knowledge, this is the first work to verify experimentally a switched adaptive control framework within the context of aerial robotics.

5.2 Mathematical preliminaries

5.2.1 Notation

In this section, we establish some of the notation used in this chapter. Let $\mathbb{N}$ denote the *set of positive integers*, $\mathbb{R}$ denote the *set of real numbers*, $\mathbb{C}$ the *set of complex numbers*, $\mathbb{R}^n$ the *set of* $n \times 1$ *real column vectors*, $\mathbb{R}^{n\times m}$ the *set of* $n \times m$ *real matrices*, $\mathcal{B}_\varepsilon(x)$ the *open ball centered* at $x \in \mathbb{R}^n$ with *radius* $\varepsilon > 0$, and $\partial\mathcal{B}_\varepsilon(x)$ the *sphere centered* at $x \in \mathbb{R}^n$ with *radius* ε. The *indicator function* of the set $\mathcal{A} \subset \mathbb{R}^n$ is denoted by $\mathbf{1}_\mathcal{A} : \mathcal{A} \to \{0, 1\}$ and is defined so that if $x \in \mathcal{A}$, then $\mathbf{1}_\mathcal{A}(x) = 1$, and if $x \notin \mathcal{A}$, then $\mathbf{1}_\mathcal{A}(x) = 0$. The *Lebesgue measure* of a set $\mathcal{S} \subset \mathbb{R}^{n\times m}$ is denoted by $\mu(\mathcal{S})$, and integrals are in the sense of Lebesgue. A property $\mathfrak{P}$ is verified *almost everywhere* with respect to the Lebesgue measure $\mu(\cdot)$ on a set $\mathcal{X} \subseteq \mathbb{R}^n$ if there exists $\mathcal{N} \subset \mathcal{X}$ such that $\mu(\mathcal{N}) = 0$ and $\mathfrak{P}$ is verified by all $x \in \mathcal{X} \setminus \mathcal{N}$. In this case, we write $\mathfrak{P}$ is verified for $x \in \mathcal{X}$ a.e..

The ith *vector of the canonical basis of* $\mathbb{R}^n$ is denoted by $\mathbf{e}_{i,n}$. The *zero vector* in $\mathbb{R}^n$ is denoted by 0_n or 0, the *zero* $n \times m$ *matrix* in $\mathbb{R}^{n\times m}$ is denoted by $0_{n\times m}$ or 0, and the *identity matrix* in $\mathbb{R}^{n\times n}$ is denoted by I_n or I. The diagonal matrix, whose diagonal entries are given by the components of $z \in \mathbb{R}^n$, is denoted by $\mathrm{diag}(z)$. The *transpose of* $B \in \mathbb{R}^{n\times m}$ is denoted by B^{T}, the *rank of* B is denoted by $\mathrm{rank}(B)$, and the *trace of* $A \in \mathbb{R}^{n\times n}$ is denoted by $\mathrm{tr}(A)$. The *spectrum* of $A \in \mathbb{R}^{n\times n}$ is denoted by $\mathrm{spec}(A)$, and the eigenvalues of A with minimum real part are denoted by $\lambda_{\min}(A)$. We write $\|\cdot\|$ for the *Euclidean vector norm* and

the corresponding *equi-induced matrix norm*. Furthermore, we write $\|\cdot\|_{\mathrm{F}}$ for the *Frobenius matrix norm*. The *Kronecker product of* $A \in \mathbb{R}^{n\times m}$ *and* $B \in \mathbb{R}^{l\times p}$ is denoted by $A \otimes B$.

5.2.2 Fundamentals of switched dynamical systems – Carathéodory framework

In the following, we recall fundamental properties of the nonlinear dynamical system with time-dependent switching

$$\dot{x}(t) = f_{\sigma(t)}(t, x(t)), \qquad x(t_0) = x_0, \qquad t \geq t_0, \tag{5.1}$$

where $f_s : [t_0, \infty) \times \mathcal{D} \to \mathbb{R}^n$, $s \in \Sigma$, $\Sigma \subset \mathbb{N}$ is bounded and denotes the *set of switching indexes*, the set $\mathcal{D} \subseteq \mathbb{R}^n$ is open, connected, and such that $0 \in \mathcal{D}$, the *switching signal* $\sigma : [t_0, \infty) \to \Sigma$ is piece-wise constant, $\sigma(t) = \lim_{\tau \to t^+} \sigma(\tau)$ for each $t \geq t_0$, the *sth dynamical model* $f_s(\cdot, x)$ is piece-wise continuous in t for all $(s, x) \in \Sigma \times \mathcal{D}$, $f_s(t, 0) = 0$ for all $(s, t) \in \Sigma \times [t_0, \infty)$, and $f_s(t, \cdot)$ is Lipschitz continuous in x uniformly in t for all t in compact subsets of $[t_0, \infty)$ and for all $s \in \Sigma$; we recall that $\sigma(\cdot)$ is piece-wise constant if and only if $\sigma(\cdot)$ has a finite number of points of discontinuity on any compact subset of $[t_0, \infty)$ and is constant between two consecutive points of discontinuity. In this paper, we define *switching times* as the points of discontinuity of $\sigma(\cdot)$.

Definition 5.1 ([184, p. 10])**.** A function $x : [t_0, \infty) \to \mathcal{D}$ is a *Carathéodory solution of* (5.1) if

$$x(t) = x_0 + \int_{t_0}^{t} f_{\sigma(\tau)}(\tau, x(\tau))\mathrm{d}\tau \qquad t \geq t_0 \ \text{ a.e..} \tag{5.2}$$

It is worthwhile to recall that if $x(\cdot)$ verifies (5.2), then $x(\cdot)$ is absolutely continuous [154, p. 128]. Furthermore, Lipschitz continuity of $f_s(t, \cdot)$ for all t in compact subsets of $[t_0, \infty)$ and for all $s \in \Sigma$ is sufficient to guarantee the existence of a unique solution of (5.1) in the

sense of Carathéodory [147, Th. 1.2]. Next, we recall the notion of uniform boundedness of Carathéodory solutions of nonlinear dynamical systems under time-dependent switching.

Definition 5.2. The switched dynamical system (5.1) is *bounded uniformly in both* $t_0 \in [0,\infty)$ *and* $\sigma(\cdot)$ if there exists $\gamma > 0$, which is independent of both $t_0 \in [0,\infty)$ and $\sigma(\cdot)$, such that for every $\delta \in (0,\gamma)$, there exists $\varepsilon(\delta) > 0$ such that $x_0 \in \mathcal{B}_\delta(0) \cap \mathcal{D}$ implies that $\|x(t)\| < \varepsilon$, $t \geq t_0$ a.e.. The switched dynamical system (5.1) is *globally bounded uniformly in both* $t_0 \in [0,\infty)$ *and* $\sigma(\cdot)$ if $\mathcal{D} = \mathbb{R}^n$ and for every $\delta > 0$ there exists $\varepsilon(\delta) > 0$ such that $x_0 \in \mathcal{B}_\delta(0)$ implies that $\|x(t)\| < \varepsilon$, $t \geq t_0$ a.e..

5.2.3 Fundamentals of switched dynamical systems – Filippov framework

In the following, we recall fundamental notions concerning Filippov solutions of the nonlinear dynamical system with time-dependent switching (5.1), where the sth dynamical model $f_s(\cdot,x)$, $s \in \Sigma$, is Lebesgue integrable and essentially locally bounded uniformly in $t \in [t_0,\infty)$ for all $(s,x) \in \Sigma \times \mathcal{D}$, $f_s(t,0) = 0$ for all $(s,t) \in \Sigma \times [t_0,\infty)$, and $f_s(t,\cdot)$ is continuous in x uniformly in t for all t in compact subsets of $[t_0,\infty)$ and for all $s \in \Sigma$, $\mathcal{D} \subseteq \mathbb{R}^n$ is open, connected, convex, and such that $0 \in \mathcal{D}$, the set of switching indexes $\Sigma \subset \mathbb{N}$ is bounded, and the switching signal $\sigma : [t_0,\infty) \to \Sigma$ is Lebesgue integrable and has countably many discontinuities. It is worthwhile to note that, while employing the Filippov framework, we do not assume that $\sigma(\cdot)$ has a finite number of switching times on compact subsets of $[t_0,\infty)$.

Definition 5.3 ([147, p. 85])**.** Let $\mathcal{I} \subseteq [t_0,\infty)$ be connected and such that $t_0 \in \mathcal{I}$. If $x : \mathcal{I} \to \mathcal{D}$ is absolutely continuous and such that

$$\dot{x}(t) \in K[f_{\sigma(t)}](t,x(t)), \qquad t \in \mathcal{I} \ \text{ a.e.}, \tag{5.3}$$

where

$$K[f_s](t,x) \triangleq \bigcap_{\delta>0} \bigcap_{\mu(\mathcal{N})=0} \overline{\text{co}}\left(f_s\left(t, \mathcal{B}_\delta(x)\setminus\mathcal{N}\right)\right), \quad (s,t,x) \in \Sigma \times [t_0,\infty) \times \mathcal{D}, \tag{5.4}$$

denotes the *Filippov regularization of* (5.1), $\bigcap_{\mu(\mathcal{N})=0}$ denotes the intersection over sets $\mathcal{N}$ of measure zero, and $\overline{\text{co}}(\cdot)$ denotes the convex closure of its argument, then $x(\cdot)$ is a *Filippov solution of* (5.1). If there do not exist a connected set $\overline{\mathcal{I}} \subseteq [t_0,\infty)$ and a Filippov solution $\overline{x} : \overline{\mathcal{I}} \to \mathcal{D}$ of (5.1) such that $\mathcal{I} \subset \overline{\mathcal{I}}$ and $\overline{x}(t) = x(t)$, $t \in \mathcal{I}$ a.e., then $x : \mathcal{I} \to \mathcal{D}$ is a *maximal Filippov solution* of (5.1).

It follows from Theorem 2.7 of [147], the boundedness of Σ, the integrability and the essential local boundedness of $f_s(\cdot,x)$, $s \in \Sigma$, uniformly in $t \in [t_0,\infty)$ for all $(s,x) \in \Sigma \times \mathcal{D}$, and the continuity of $f_s(t,\cdot)$ in x uniformly in t for all t in compact subsets of $[t_0,\infty)$ and for all $s \in \Sigma$, that there exists a solution of (5.1) in the sense of Filippov. Next, we recall the notions of directional derivatives, generalized directional derivatives, and regular functions. For the statement of these definitions, let $[z, z+a) \triangleq \{z+\theta a, \theta \in [0,1)\}$, $(z,a) \in \mathbb{R}^l \times \mathbb{R}^l$, denote a *line segment in* $\mathbb{R}^l$ and let

$$\text{vcone}(\mathcal{Q},z) \triangleq \left\{\xi \in \mathbb{R}^l : \exists \alpha > 0 \,\text{such that}\, [z, z+\alpha\xi) \subset \mathcal{Q}\right\} \tag{5.5}$$

denote the *variational cone* of $\mathcal{Q} \subseteq \mathbb{R}^l$ at z.

Definition 5.4 ([185, pp. 63-64],[186, p. 39])**.** Let $W : \mathcal{Q} \to \mathbb{R}$ be Lipschitz continuous, where $\mathcal{Q} \subseteq \mathbb{R}^l$. The *right directional derivative of* $W(\cdot)$ *at* $z \in \mathcal{Q}$ *along the direction of* $q \in \text{vcone}(\mathcal{Q},z)$ is defined as

$$W'(z,q) \triangleq \lim_{\tau \to 0^+} \frac{W(z+\tau q) - W(z)}{\tau}, \qquad (z,q) \in \mathcal{Q} \times \text{vcone}(\mathcal{Q},z). \tag{5.6}$$

The *generalized directional derivatives of* $W(\cdot)$ *at* $z \in \mathcal{Q}$ *along the direction of* $q \in \text{vcone}(\mathcal{Q},z)$

is defined as

$$W^0(z,q) \triangleq \limsup_{\substack{y\to z\\ \tau\to 0^+}} \frac{W(y+\tau q)-W(y)}{\tau}, \qquad (z,q)\in\mathcal{Q}\times \mathrm{vcone}(\mathcal{Q},z). \tag{5.7}$$

If $W'(z,q)=W^0(z,q)$ for all $q\in\mathrm{vcone}(\mathcal{Q},z)$, then $W(\cdot)$ is *regular at* $z\in\mathcal{Q}$.

Next, we recall the notion of Clarke gradient. This definition is essential to state a generalization of the LaSalle-Yoshizawa theorem for Lebesgue measurable dynamical models.

Definition 5.5 ([186, p. 10])**.** Let $W:\mathcal{Q}\to\mathbb{R}$, where $\mathcal{Q}\subseteq\mathbb{R}^l$. The *Clarke gradient* of $W(\cdot)$ at $z\in\mathcal{Q}$ is defined as

$$\partial W(z) \triangleq \left\{p\in\mathbb{R}^l : W^0(z,q)\leq p^{\mathrm{T}}q, \forall q\in\mathrm{vcone}(\mathcal{Q},z)\right\}, \qquad z\in\mathcal{Q}, \tag{5.8}$$

where $W^0(\cdot,\cdot)$ denotes the generalized directional derivatives of $W(\cdot)$.

In the following, we provide an expression of the Clarke gradient for Lipschitz continuous functions. For the statement of this result, recall that, by Rademacher's theorem [187, Th. 3.1.6], if $V:[t_0,\infty)\times\mathcal{D}\to\mathbb{R}$ is Lipschitz continuous, then $V(\cdot,\cdot)$ is differentiable almost everywhere, and define

$$\Omega_V \triangleq \left\{(t,x)\in[t_0,\infty)\times\mathcal{D} : \left[\frac{\partial V(t,x)}{\partial t}, \frac{\partial V(t,x)}{\partial x}\right]^{\mathrm{T}} \text{is not defined}\right\} \tag{5.9}$$

as the set wherein $V(\cdot,\cdot)$ is not differentiable.

Theorem 5.6 ([186, p. 63])**.** *Let $V:[t_0,\infty)\times\mathcal{D}\to\mathbb{R}$ be Lipschitz continuous. The Clarke gradient of $V(\cdot,\cdot)$ at $(t,x)\in[t_0,\infty)\times\mathcal{D}$ is given by*

$$\partial V(t,x) = \overline{\mathrm{co}}\Bigg\{\lim_{i\to\infty}\left[\frac{\partial V(t_i,x_i)}{\partial t}, \frac{\partial V(t_i,x_i)}{\partial x}\right]^{\mathrm{T}} : (t_i,x_i)\to(t,x),\\ (t_i,x_i)\notin\Omega_V, x_i\notin\mathcal{N}, i\in\mathbb{N}\Bigg\}, \qquad (t,x)\in[t_0,\infty)\times\mathcal{D}, \tag{5.10}$$

where $\mathcal{N} \subset \mathcal{D}$ is an arbitrary set of measure zero.

Next, we recall a result that characterizes the total derivative of a Lipschitz continuous, regular function. For the statement of this result, consider the nonlinear, time-varying dynamical system

$$\dot{x}(t) = f(t, x(t)), \qquad x(t_0) = x_0, \qquad t \geq t_0, \tag{5.11}$$

where $f : [t_0, \infty) \times \mathcal{D} \to \mathbb{R}^n$ is such that $f(\cdot, x)$ is Lebesgue integrable and essentially locally bounded uniformly in $t \in [t_0, \infty)$ for all $x \in \mathcal{D}$ and $f(t, \cdot)$ is continuous in x uniformly in t for all t in compact subsets of $[t_0, \infty)$.

Lemma 5.7 ([167])**.** *Let $x : [t_0, \infty) \to \mathcal{D}$ denote a solution of (5.11) in the sense of Filippov and let $V : [t_0, \infty) \times \mathcal{D} \to \mathbb{R}$ be Lipschitz continuous and regular. Then $V(t, x(t))$, $t \geq t_0$, is absolutely continuous, $\dot{V}(t, x(t))$ exists almost everywhere on $[t_0, \infty)$, and $\dot{V}(t, x(t)) \in \dot{\overline{V}}(t, x(t))$, $t \in [t_0, \infty)$ a.e., where*

$$\dot{\overline{V}}(t, x) \triangleq \bigcap_{\xi \in \partial V(t,x)} \xi^{\mathrm{T}} \begin{bmatrix} K[f](t, x) \\ 1 \end{bmatrix}, \qquad (t, x) \in [t_0, \infty) \times \mathcal{D}. \tag{5.12}$$

The next result, which is a direct consequence of Corollary 1 of [167], guarantees that maximal solutions of (5.1) in the sense of Filippov are defined on $\mathcal{I} = [t_0, \infty)$, and provides a generalization of the LaSalle-Yoshizawa theorem [117, Th. 4.7]. For the statement of this result, let $f : [t_0, \infty) \times \mathcal{D} \to \mathbb{R}^n$ be so that $f_{\sigma(t)}(t, x) = f(t, x)$, $(t, x) \in [t_0, \infty) \times \mathcal{D}$, and the nonlinear differential equation under time-dependent switching (5.1) is equivalent to the nonlinear, time-varying, discontinuous dynamical system (5.11).

Theorem 5.8. *Consider the nonlinear, discontinuous dynamical systems (5.1) and (5.11). Let $r > 0$ be such that $\mathcal{B}_r(0) \subset \mathcal{D}$, let $V : [t_0, \infty) \times \mathcal{D} \to \mathbb{R}$ be Lipschitz continuous and regular, let $W_1, W_2, W_3 : \mathcal{D} \to \mathbb{R}$ be such that both $W_1(\cdot)$ and $W_2(\cdot)$ are positive-definite and*

$W_3(\cdot)$ is nonnegative-definite, and let $c \in \big(0, \min_{\partial \mathcal{B}_r(0)} W_1(x)\big)$. If

$$W_1(x) \leq V(t,x) \leq W_2(x), \qquad (t,x) \in [t_0, \infty) \times \mathcal{D}, \tag{5.13}$$

$$\dot{V}(t, x(t)) \leq -W_3(x(t)), \qquad t \geq t_0 \ \text{ a.e.}, \tag{5.14}$$

where $x(\cdot)$ denotes a maximal solution of (5.11) in the sense of Filippov such that $x(t_0) \in \{x \in \mathcal{B}_r(0) : W_2(x) \leq c\}$, then $x : [t_0, \infty) \to \mathcal{D}$ is bounded and such that $\lim_{t\to\infty} W_3(x(t)) = 0$. Furthermore, if $\mathcal{D} = \mathbb{R}^n$ and both $W_1(\cdot)$ and $W_2(\cdot)$ are radially unbounded, then every maximal solution $x(\cdot)$ of the Filippov regularization of (5.11) is bounded uniformly in both $t_0 \in [0, \infty)$ and $\sigma(\cdot)$, and such that $\lim_{t\to\infty} W_3(x(t)) = 0$ for all $x_0 \in \mathbb{R}^n$ uniformly in both t_0 and $\sigma(\cdot)$.

It is worthwhile to note that Theorem 5.8 does not involve any condition on the dwell-time of the nonlinear dynamical system (5.1), that is, on the minimal time interval between any pair of consecutive switching times [184, p. 56], but relies on the assumption that the switching signal $\sigma(\cdot)$ has countably many discontinuities. As discussed in Remark 1 of [168], if $\sigma(\cdot)$ is Lebesgue measurable, but does not have countably many discontinuities, then Theorem 2 of [168] provides an alternative to Theorem 5.8 above.

5.3 Model reference adaptive control of switched dynamical systems

5.3.1 Problem formulation

In this section, we design an adaptive control law for unknown nonlinear plants, whose dynamics are captured by time-dependent switching among multiple models, so that their trajectories mimic the trajectories of user-defined reference models under time-dependent

switching. Specifically, consider the nonlinear *plant* under time-dependent switching

$$\dot{x}(t) = A_{\sigma(t)}x(t) + B_{\sigma(t)}\left[u(t) + \Theta^{\mathrm{T}}\Phi_{\sigma(t)}(t, x(t))\right], \quad x(t_0) = x_0, \quad t \geq t_0, \tag{5.15}$$

where $x(t) \in \mathcal{D}$, $t \geq t_0 \geq 0$, denotes the *plant's trajectory*, $u(t) \in \mathbb{R}^m$ denotes the *control input*, the set $\mathcal{D} \subseteq \mathbb{R}^n$ is open, connected, and such that $0 \in \mathcal{D}$, $\sigma : [t_0, \infty) \to \Sigma$ denotes the switching signal and is user-defined, $\Sigma \subset \mathbb{N}$ is bounded, $A_s \in \mathbb{R}^{n\times n}$ is unknown, $s \in \Sigma$, $B_s \in \mathbb{R}^{n\times m}$ is known, $\Theta \in \mathbb{R}^{N\times m}$ is unknown, the *regressor vector* $\Phi_s : [t_0, \infty) \times \mathbb{R}^n \to \mathbb{R}^N$ is known, Lebesgue integrable, and jointly continuous in its arguments, and $\Phi_s(t, \cdot)$ is Lipschitz continuous in x uniformly in t on compact subsets of $[t_0, \infty)$. Without loss of generality, we assume that Σ comprises the first $\overline{\sigma}$ positive integers, where $\overline{\sigma}$ denotes the cardinality of Σ.

The unknown matrix A_s, $s \in \Sigma$, in (5.15) captures *parametric uncertainties*, and the mapping $s \mapsto A_s$ is considered as unknown. We assume that the pairs (A_s, B_s) are controllable for all $s \in \Sigma$; although the entries of A_s are unknown, this hypothesis can be verified in problems of practical interest since the structure of A_s is usually known [75, p. 281].

The term $\Theta^{\mathrm{T}}\Phi_s(t, x)$, $(s, t, x) \in \Sigma\times[t_0, \infty)\times\mathbb{R}^n$, in (5.15) captures *matched uncertainties*. Matched uncertainties may be equivalently captured by $\bar{\Theta}^{\mathrm{T}}_{\sigma(t)}\ \bar{\Phi}_{\sigma(t)}(t, x)$, $(t, x) \in [t_0, \infty) \times \mathbb{R}^n$, where $\bar{\Theta}_s \in \mathbb{R}^{\bar{N}_s\times m}$, $s \in \Sigma$, is unknown, the mapping $s \mapsto \bar{\Theta}_s$ is unknown, and $\bar{\Phi}_s : [t_0, \infty) \times \mathbb{R}^n \to \mathbb{R}^{\bar{N}_s}$ is known. However, there always exist $\Theta \in \mathbb{R}^{N\times m}$ and a *regressor vector* $\Phi_s : [t_0, \infty) \times \mathbb{R}^n \to \mathbb{R}^N$, $s \in \Sigma$ such that $\Theta^{\mathrm{T}}\Phi_{\sigma(t)}(t, x) = \bar{\Theta}^{\mathrm{T}}_{\sigma(t)}\bar{\Phi}_{\sigma(t)}(t, x)$, $t \geq t_0$. Indeed, let $\Sigma_1, \ldots, \Sigma_p \subseteq \Sigma$, $p \leq \overline{\sigma}$, denote partitions of Σ. Uncertainties in the mapping $s \mapsto \bar{\Theta}_s$ can be captured by designing Σ_j, $j = 1, \ldots, p$, as a non-singleton set, and the dynamical model (5.15) can be deduced by setting $\Phi_s(t, x) = \left[\mathbf{1}_{\Sigma_1}(s)\bar{\Phi}^{\mathrm{T}}_1(t, x), \ldots, \mathbf{1}_{\Sigma_p}(s)\bar{\Phi}^{\mathrm{T}}_p(t, x)\right]^{\mathrm{T}}$, $(s, t, x) \in \Sigma \times [t_0, \infty) \times \mathbb{R}^n$, $\Theta = \left[\bar{\Theta}^{\mathrm{T}}_1, \ldots, \bar{\Theta}^{\mathrm{T}}_p\right]^{\mathrm{T}}$, and $N = \sum_{j=1}^{p} \bar{N}_j$. Regressor vectors are usually designed leveraging on prior knowledge of the plant dynamics [75, Ch. 9].

Consider also the *reference dynamical model* under time-dependent switching

$$\dot{x}_{\rm ref}(t) = A_{{\rm ref},\sigma(t)}x_{\rm ref}(t) + B_{{\rm ref},\sigma(t)}r(t), \qquad x_{\rm ref}(t_0) = x_{{\rm ref},0}, \qquad t \geq t_0, \tag{5.16}$$

where the $x_{\rm ref}(t) \in \mathbb{R}^n$, $t \geq t_0$, denotes the *reference trajectory*, the *reference command input* $r(t) \in \mathbb{R}^m$ is piece-wise continuous and bounded, $A_{{\rm ref},s} \in \mathbb{R}^{n\times n}$ is Hurwitz, $s \in \Sigma$, and $B_{{\rm ref},s} \in \mathbb{R}^{n\times m}$, and assume there exist pairs $(K_{x,s}, K_{r,s}) \in \mathbb{R}^{n\times m} \times \mathbb{R}^{m\times m}$ such that the *matching conditions*

$$A_{{\rm ref},s} = A_s + B_s K_{x,s}^{\rm T}, \qquad s \in \Sigma, \tag{5.17}$$

$$B_{{\rm ref},s} = B_s K_{r,s}^{\rm T}, \tag{5.18}$$

are verified. In the following, we show that the reference dynamical model (5.16) is input-to-state stable.

Theorem 5.9. *Consider the dynamical system* (5.16) *under time-dependent switching. If the shortest time interval in which* $\sigma(\cdot)$ *remains constant is greater than some minimum dwell time,* $t_{\rm d} > 0$*, then* (5.16) *is input-to-state stable and globally bounded uniformly in* $t_0 \in [0, \infty)$*. Furthermore, if* $r \equiv 0$*,* $t \geq t_0$*, then* (5.16) *is globally asymptotically stable uniformly in* $t_0 \in [0, \infty)$ *for any* $\sigma(\cdot)$ *that verifies the dwell time condition.*

Proof: Through induction, it can be shown that

$$\begin{aligned}
\|x_{\rm ref}(t)\| &\leq \prod_{j=0}^{n_{\rm d}(t,t_0)} \gamma_{\sigma(\tau_j)} \|x_{\rm ref}(t_0)\| e^{-\beta_{\min}(t-t_0)} \\
&\quad + \left[1 + \sum_{j=1}^{n_{\rm d}(t,t_0)} \prod_{k=j}^{n_{\rm d}(t,t_0)} \gamma_{\sigma(\tau_k)} e^{-\beta_{\min}(t-\tau_j)}\right] \frac{\gamma_{\max}}{\beta_{\min}} \max_{s\in\Sigma} \|B_{\rm ref,s}\| \sup_{\mu\in[t_0,t]} \|r(\mu)\|, \\
&= \prod_{j=0}^{n_{\rm d}(t,t_0)} \gamma_{\sigma(\tau_j)} \|x_{\rm ref}(t_0)\| e^{-\beta_{\min}(t-t_0)}
\end{aligned} \tag{5.19}$$

$$+\left[1+\gamma_{\sigma(\tau_{n_{\mathrm{d}}(t,t_0)})}e^{-\beta_{\min}(t-\tau_{n_{\mathrm{d}}}(t,t_0))}+\sum_{j=1}^{n_{\mathrm{d}}(t,t_0)-1}\prod_{k=j}^{n_{\mathrm{d}}(t,t_0)-1}\gamma_{\sigma(\tau_k)}e^{-\beta_{\min}(\tau_{\mathrm{d}}-\tau_j)}\right]$$
$$\cdot\frac{\gamma_{\max}}{\beta_{\min}}\max_{s\in\Sigma}\|B_{\mathrm{ref,s}}\|\sup_{\mu\in[t_0,t]}\|r(\mu)\|, \tag{5.20}$$
$$\leq\prod_{j=0}^{n_{\mathrm{d}}(t,t_0)}\gamma_{\sigma(\tau_j)}\|x_{\mathrm{ref}}(t_0)\|e^{-\beta_{\min}(t-t_0)}$$
$$+\left[1+\gamma_{\sigma(\tau_{n_{\mathrm{d}}(t,t_0)})}+\sum_{j=1}^{n_{\mathrm{d}}(t,t_0)-1}\prod_{k=j}^{n_{\mathrm{d}}(t,t_0)-1}\gamma_{\sigma(\tau_k)}e^{-\beta_{\min}(\tau_{\mathrm{d}}-\tau_j)}\right]$$
$$\cdot\frac{\gamma_{\max}}{\beta_{\min}}\max_{s\in\Sigma}\|B_{\mathrm{ref,s}}\|\sup_{\mu\in[t_0,t]}\|r(\mu)\|. \tag{5.21}$$

Recall that the system's dwell time is assumed to be strictly greater than 0, and hence, the exponential term in (5.21) is always less than 1. Let $\varepsilon \triangleq e^{-\beta_{\min}t_{\mathrm{d}}}$, where t_{d} denotes the *dwell time*, and since instantaneous switching is not allowed, $\varepsilon \in (0,1)$. Consider the less conservative estimate of the sequence in (5.21) as

$$\sum_{j=1}^{n_{\mathrm{d}}(t,t_0)-1}\prod_{k=j}^{n_{\mathrm{d}}(t,t_0)-1}\gamma_{\sigma(\tau_k)}\varepsilon, \tag{5.22}$$

let $\gamma_{\max}=\max_{s\in\Sigma}\gamma_s$, and consider the case for arbitrary switching, hence $n_{\mathrm{d}}(\cdot,\cdot)$ becomes arbitrarily large, then

$$\sum_{j=1}^{n_{\mathrm{d}}(t,t_0)-1}\prod_{k=j}^{n_{\mathrm{d}}(t,t_0)-1}\gamma_{\max}\varepsilon=\lim_{t\to\infty}\left(\sum_{j=1}^{n_{\mathrm{d}}(t,t_0)-1}(\gamma_{\max}\varepsilon)^{n_{\mathrm{d}}(t,t_0)-1-j}\right)=\lim_{t\to\infty}\left(\frac{(\gamma_{\max}\varepsilon)^{n_{\mathrm{d}}(t,t_0)}-(\gamma_{\max}\varepsilon)}{\gamma_{\max}\varepsilon-1}\right),$$
$$=\begin{cases}\frac{\gamma_{\max}\varepsilon}{1-\gamma_{\max}\varepsilon}, & \gamma_{\max}\varepsilon<1,\\ \infty, & \gamma_{\max}\varepsilon\geq 1,\end{cases} \tag{5.23}$$

where this sequence is finite if $\gamma_{\max}\varepsilon<1$, and by the definition of ε, the minimum dwell time such that the switched reference system is input-to-state stable is given by $t_{\mathrm{d}}>\frac{\ln\gamma_{\max}}{\beta_{\min}}$. The same logic can be applied to the first product in (5.21) which converges with the same

condition on the dwell time. Since $r(\cdot)$ is bounded by assumption, it follows that (5.16) is globally bounded uniformly in $t_0 \in [0, \infty)$. Lastly, if $r \equiv 0$, $t \geq t_0$, then it follows that (5.16) is globally asymptotically stable uniformly in $t_0 \in [0, \infty)$. It is worthwhile to note that similar results have been found for single-input single-output systems in [188] and by studying the Lyapunov functions of switched dynamical systems in [189]. ■

Corollary 5.10. *Consider the switched linear dynamical system (17) under arbitrary switching, and assume there exists a common quadratic Lyapunov function for the matrices* $A_{\mathrm{ref,s}}$, $\forall s \in \Sigma$. *Then, the switched dynamical system is input-to-state stable and globally bounded uniformly in* $t_0 \in [0, \infty)$ *for any* $\sigma(\cdot)$ *with dwell time,* $t_{\mathrm{d}} > 0$.

Proof: It is known that if a common quadratic Lyapunov function exists for the matrices $A_{\mathrm{ref,s}}$, $\forall s \in \Sigma$, then the uncontrolled system is exponentially stable for any switching law [184, Theorem 2.1]. It follows that for (13) to be exponentially stable for any $\sigma(\cdot)$ when $r(\cdot) = 0$, $t \geq t_0$, then $\gamma_s = 1$, $\forall s \in \Sigma$, since

$$\|e^{A_{\mathrm{ref,s}}t}\|\|x_0\| \leq \gamma_s e^{-\beta_s t}\|x_0\|, \tag{5.24}$$

$$\|e^{A_{\mathrm{ref,s}}t}\| \leq \gamma_s e^{-\beta_s t}, \tag{5.25}$$

where at $t = 0$, $\|I\| \leq \gamma_s$. In this case, (5.22) becomes $\sum_{j=1}^{n_{\mathrm{d}}(t,t_0)-1} \prod_{k=j}^{n_{\mathrm{d}}(t,t_0)-1} \varepsilon$, and the solution by applying (5.23) is

$$\sum_{j=1}^{n_{\mathrm{d}}(t,t_0)-1} \prod_{k=j}^{n_{\mathrm{d}}(t,t_0)-1} \varepsilon = \frac{\epsilon}{1-\epsilon}, \tag{5.26}$$

which is finite for any $\epsilon \in (0, 1)$, or equivalently $t_{\mathrm{d}} > 0$. Since $r(\cdot)$ is bounded by assumption, it follows from (32) that (17) is globally bounded uniformly in $t_0 \in [0, \infty)$ for any $\sigma(\cdot)$. Similar results were proved for single-input single-output linear systems in [188] and for nonlinear systems using Lyapunov methods in [190]. ■

As an example, consider an orthogonal decomposition of the matrix $A_{\text{ref,s}}$ such that

$$A_{\text{ref},s} = S_s^{-1} \begin{bmatrix} \lambda_{1,s} & 0 \\ 0 & \lambda_{2,s} \end{bmatrix} S_s, \tag{5.27}$$

where $\lambda_{1,s}, \lambda_{2,s} < 0$ are the eigenvalues of $A_{\text{ref,s}}$, and S_s is such that $S_s^{-1} = S_s^{\mathrm{T}}$ and $\|S_s\| = 1$. The spectral theorem states that all symmetric matrices are orthogonal diagonalizable in this form [191, Theorem 12.22]. It can be shown that for a given τ_j, the factors, $\gamma_{\sigma(\tau_j)}$ in (5.16), can be found as $\gamma_{\sigma(\tau_j)} = \|S_s^{-1}\|\|S_s\|$. In this case, it holds that $\text{spec}(A_{\text{ref},s}) = \{\lambda_{1,s}, \lambda_{2,s}\}$, $s \in \Sigma$, $\gamma_s = 1$ since S_s is orthogonal, and $\beta_s = \max_{s\in\Sigma}(\lambda_{1,s}, \lambda_{2,s})$. Thus, the conditions of Theorem 5.9 are verified since $\sum_{j=0}^{m} \log(\gamma_{\sigma(\tau_j)}) = 0$ and $\sum_{j=0}^{m} \Pi_{k=j}^{\infty} \gamma_{\sigma(\tau_j)} = 0$ for any switching signal and hence, both $\left\{\sum_{j=0}^{m} \log(\gamma_{\sigma(\tau_j)})\right\}_{m=0}^{\infty}$ and $\left\{\sum_{j=0}^{m} \Pi_{k=j}^{\infty} \gamma_{\sigma(\tau_j)}\right\}_{m=0}^{\infty}$ are Cauchy sequences.

Lastly, assume there exists a symmetric positive-definite matrix $P \in \mathbb{R}^{n\times n}$ that verifies the set of Lyapunov matrix inequalities

$$A_{\text{ref},s}^{\mathrm{T}} P + P A_{\text{ref},s} < 0, \qquad s \in \Sigma, \tag{5.28}$$

and consider the *trajectory tracking error dynamics*

$$\dot{e}(t) = A_{\text{ref},\sigma(t)} e(t) + B_{\sigma(t)} \Delta\Theta^{\mathrm{T}}(t) \tilde{\Phi}_{\sigma(t)}(t, x(t)), \;\; e(t_0) = x_0 - x_{\text{ref},0}, \;\; t \geq t_0, \tag{5.29}$$

and the *adaptive law*

$$\dot{\hat{\Theta}}(t) = -\Gamma \tilde{\Phi}_{\sigma(t)}(t, x(t)) e^{\mathrm{T}}(t) P B_{\sigma(t)}, \qquad \hat{\Theta}(t_0) = \hat{\Theta}_0, \tag{5.30}$$

where $\hat{\Theta} : [t_0, \infty) \to \mathbb{R}^{(\overline{\sigma}(n+m)+N)\times m}$ denotes the *adaptive gain*, $\Delta\Theta(t) \triangleq \hat{\Theta}(t) - \tilde{\Theta}$,

$$\tilde{\Phi}_s(t,x) \triangleq \begin{bmatrix} \mathcal{I}(s) \otimes x \\ \mathcal{I}(s) \otimes r(t) \\ -\Phi_s(t,x) \end{bmatrix}, \qquad (s,t,x) \in \Sigma \times [t_0,\infty) \times \mathbb{R}^n, \tag{5.31}$$

$$\mathcal{I}(s) \triangleq \left[\mathbf{1}_{\{s\in\Sigma:s-1=0\}}(s), \ldots, \mathbf{1}_{\{s\in\Sigma:s-\overline{\sigma}=0\}}(s)\right]^{\mathrm{T}}, \tag{5.32}$$

$$\tilde{\Theta} \triangleq \left[K_{x,1}^{\mathrm{T}}, \ldots, K_{x,\overline{\sigma}}^{\mathrm{T}}, K_{r,1}^{\mathrm{T}}, \ldots, K_{r,\overline{\sigma}}^{\mathrm{T}}, \Theta^{\mathrm{T}}\right]^{\mathrm{T}}, \tag{5.33}$$

$x(\cdot)$ verifies (5.15) with $u(t) = \phi(\hat{\Theta}(t), \tilde{\Phi}_{\sigma(t)}(t, x(t)))$,

$$\phi(\hat{\Theta}, \tilde{\Phi}_s) = \hat{\Theta}^{\mathrm{T}}\tilde{\Phi}_s, \qquad (s, \hat{\Theta}, \tilde{\Phi}_s) \in \Sigma \times \mathbb{R}^{(\overline{\sigma}(n+m)+N)\times m} \times \mathbb{R}^{\overline{\sigma}(n+m)+N}, \tag{5.34}$$

denotes the *control law*, and $\Gamma \in \mathbb{R}^{(\overline{\sigma}(n+m)+N)\times(\overline{\sigma}(n+m)+N)}$ is symmetric and positive-definite, and denotes the *adaptive rate matrix*.

Our goal is to prove that solutions of both (5.29) and (5.30) are bounded uniformly in $t_0 \in [0, \infty)$ and $\sigma(\cdot)$ and that solutions of (5.29) converge asymptotically to zero uniformly in t_0 and $\sigma(\cdot)$. Since both (5.29) and (5.30) are discontinuous, multiple notions of solutions may be applied [153]. Two classes of absolutely continuous [154, p. 127] generalized solutions of (5.29) and (5.30) are considered, namely Carathéodory and Filippov solutions.

In the Carathéodory framework, it can be verified that if $x(\cdot)$ denotes the solution of (5.15) with $u(t) = \phi(\hat{\Theta}(t), \tilde{\Phi}_{\sigma(t)})$, $t \geq t_0$, and $x_{\mathrm{ref}}(\cdot)$ denotes the solution of (5.16), then the solution $e(\cdot)$ of (5.29) is such that $e(t) = x(t) - x_{\mathrm{ref}}(t)$, $t \geq t_0$ a.e.. However, a solution $e(\cdot)$ of (5.29) in the sense of Filippov is not necessarily equivalent to the difference of a solution $x(\cdot)$ of (5.15) with $u(t) = \phi(\hat{\Theta}(t), \tilde{\Phi}_s(t))$, $t \geq t_0$, and a solution $x_{\mathrm{ref}}(\cdot)$ of (5.16). If $x(t) = x_{\mathrm{ref}}(t)$, $t \geq T$ a.e., for some $T \geq t_0$, and $e(t) = 0$, then this equivalence can be established for all $t \geq T$ a.e.; for details, see [152].

In the following, $n_{\mathrm{d}}(t, t_0) \in \mathbb{N}$ denotes the *number of discontinuities of* $\sigma(\cdot)$ *over the*

interval (t_0, t), $\mathcal{T}_{\sigma(t)} \triangleq \{\tau_j \in [t_0, t) : \sigma(\cdot) \text{ is discontinuous at } \tau_j, j = 0, \ldots, n_{\mathrm{d}}(t, t_0)\}$ denotes the totally ordered *set of switching times over* $[t_0, t)$, and we set $t_0 = \tau_0 \in \mathcal{T}_{\sigma(t)}$ so that both $A_{\mathrm{ref},\sigma(\cdot)}$ and $B_{\mathrm{ref},\sigma(\cdot)}$ are constant between switching times, that is, $(A_{\mathrm{ref},\sigma(\mu)}, B_{\mathrm{ref},\sigma(\mu)}) = (A_{\mathrm{ref},\sigma(\tau_j)}, B_{\mathrm{ref},\sigma(\tau_j)})$ for all $\mu \in [\tau_j, \tau_{j+1}) \cap [t_0, t)$ and for all $j = 0, \ldots, n_{\mathrm{d}}(t, t_0)$, where $\tau_j \in \mathcal{T}_{\sigma(t)}$ and $(A_{\mathrm{ref},\sigma(\tau_j)}, B_{\mathrm{ref},\sigma(\tau_j)}) = \lim_{\tau \to 0^+} (A_{\mathrm{ref},\sigma(\tau_j+\tau)}, B_{\mathrm{ref},\sigma(\tau_j+\tau)})$. For simplicity of notation, we define $\mathcal{T} \triangleq \lim_{t\to\infty} \mathcal{T}_{\sigma(t)}$.

5.3.2 Carathéodory framework

In this section, we address the model reference adaptive control design problem posed in Section 5.3.1 by analyzing Carathéodory solutions of the trajectory tracking error dynamics (5.29) and the adaptive law (5.30) and assuming that the switching signal $\sigma(\cdot)$ is piece-wise constant and such that $\sigma(t) = \lim_{\tau \to t^+} \sigma(\tau)$ for each $\tau \geq t_0$. It is worthwhile to note that the switched dynamical system given by (5.29) and (5.30) is continuous in $t \in [\tau_{j-1}, \tau_j)$ for all $(j, \tau_{j-1}, e, \hat{\Theta}) \in \mathbb{N} \times \mathcal{T} \times \mathbb{R}^n \times \mathbb{R}^{(\bar{\sigma}(n+m)+N)\times m}$ and locally Lipschitz continuous in $(e, \hat{\Theta})$ uniformly in t for all t in compact subsets of $[t_0, \infty)$, and hence it follows from Theorem 1.2 of [147] that there exists a unique pair $(e, \hat{\Theta}) : [t_0, \infty) \to \mathbb{R}^n \times \mathbb{R}^{(\bar{\sigma}(n+m)+N)\times m}$ that verifies (5.29) and (5.30) in the sense of Carathéodory.

The next theorem is the main result of this section and proves that if the trajectory tracking error $e(\cdot)$ and the adaptive gain matrix $\hat{\Theta}(\cdot)$ verify (5.29) and (5.30), respectively, in the sense of Carathéodory, then both the trajectory tracking error and the adaptive gain matrix are uniformly bounded, and the closed-loop plant's trajectory asymptotically converges to the reference model's trajectory. To prove this result, it is worthwhile to recall the following generalization of Barbalat's lemma [124, Lemma 8.2] and that the *dwell-time* $t_{\mathrm{d}} \triangleq \inf\{|\tau_j - \tau_{j-1}| : \tau_{j-1} \in \mathcal{T}, j \in \mathbb{N}\}$ of the switching signal $\sigma(\cdot)$ captures the length of the minimal interval between switching times [184, p. 56].

Lemma 5.11 ([192]). *Let* $h : [t_0, \infty) \to \mathbb{R}$ *be piece-wise continuously differentiable and*

let $\{t_k\}_{k=1}^{\infty} \subset [t_0, \infty)$ *denote the sequence of points of discontinuity of* $h(\cdot)$*. Suppose that* $\inf_{k\in\mathbb{N}} |t_k - t_{k-1}| > 0$ *and that both* $h(\cdot)$ *and* $\dot{h}(\cdot)$ *are bounded on* $[t_{k-1}, t_k)$ *uniformly in* $k \in \mathbb{N}$*. If* $\lim_{t\to\infty} \int_0^t h(\tau)\mathrm{d}\tau$ *exists and is finite, then* $\lim_{t\to\infty} h(t) = 0$ *uniformly in* $k \in \mathbb{N}$*.*

Theorem 5.12. *Consider the closed-loop trajectory tracking error dynamics* (5.29) *and the adaptive law* (5.30)*. Assume that the matching conditions* (5.17) *and* (5.18) *are verified,* $t_{\mathrm{d}} > 0$*, and there exist symmetric positive-definite matrices* $P, Q \in \mathbb{R}^{n\times n}$ *so that*

$$A_{\mathrm{ref},s}^{\mathrm{T}} P + P A_{\mathrm{ref},s} < -Q. \qquad s \in \Sigma. \tag{5.35}$$

Then, both the trajectory tracking error $e(\cdot)$ *and the adaptive gain matrix* $\hat{\Theta}(\cdot)$ *are bounded uniformly in both* $t_0 \in [0, \infty)$ *and* $\sigma(\cdot)$*, and* $e(t) \to 0$ *as* $t \to \infty$ *uniformly in both* t_0 *and* $\sigma(\cdot)$*.*

Proof: Consider the common Lyapunov function candidate

$$V(t, e, \Delta\Theta) = e^{\mathrm{T}} P e + \mathrm{tr}\left(\Delta\Theta^{\mathrm{T}} \Gamma^{-1} \Delta\Theta\right), \qquad (t, e, \Delta\Theta) \in [t_0, \infty) \times \mathbb{R}^n \times \mathbb{R}^{(\bar{\sigma}(n+m)+N)\times m}, \tag{5.36}$$

and note that if there exist symmetric positive-definite matrices $P, Q \in \mathbb{R}^{n\times n}$ so that (5.35) is verified, then the Lyapunov inequality (5.28) is verified. Next, it follows from (5.36) and (5.35) that

$$\begin{aligned}
\dot{V}(t, e(t), \Delta\Theta(t)) &\leq -\alpha_{\min}\|e(t)\|^2 + 2e^{\mathrm{T}}(t) P B_{\sigma(t)} \Delta\Theta^{\mathrm{T}}(t) \tilde{\Phi}_{\sigma(t)}(t, x(t)) \\
&\quad + 2\mathrm{tr}\left(\Delta\Theta^{\mathrm{T}}(t) \Gamma^{-1} \dot{\hat{\Theta}}(t)\right) \\
&= -\alpha_{\min}\|e(t)\|^2 \\
&\quad + 2\mathrm{tr}\left(\Delta\Theta^{\mathrm{T}}(t) \left[\Gamma^{-1} \dot{\hat{\Theta}}(t) + \tilde{\Phi}_{\sigma(t)}(t, x(t)) e^{\mathrm{T}}(t) P B_{\sigma(t)}\right]\right) \\
&= -\alpha_{\min}\|e(t)\|^2, \qquad t \geq t_0 \ \text{ a.e.},
\end{aligned} \tag{5.37}$$

along the trajectories of (5.29) and (5.30), where $\alpha_{\min} \triangleq \lambda_{\min}(Q)$.

Since both $V(\cdot,\cdot,\cdot)$ and $\dot{V}(\cdot,\cdot,\cdot)$ do not explicitly depend on t and $\sigma(\cdot)$ and $\dot{V}(t, e(t), \Delta\Theta(t))$, $t \geq t_0$, is a non-increasing function of time, by proceeding as in Theorem 4.13 of [117] it can be proven that both $e(\cdot)$ and $\hat{\Theta}(\cdot)$ are bounded on $[\tau_{j-1}, \tau_j)$ uniformly in $j \in \mathbb{N}$ for all $\tau_j \in \mathcal{T}$. Next, since $V(t, e, \Delta\Theta)$, $(t, e, \Delta\Theta) \in [t_0, \infty) \times \mathbb{R}^n \times \mathbb{R}^{(\bar{\sigma}(n+m)+N)\times m}$, is positive-definite and $\dot{V}(t, e(t), \Delta\Theta(t))$ is non-positive definite, it follows from the monotone convergence theorem [117, Th. 2.10] that there exists $V_e \geq 0$ such that $V(t, e(t), \Delta\Theta(t)) \to V_e$ as $t \to \infty$. Moreover, $\dot{x}_{\mathrm{ref}}(\cdot)$ is bounded on $[\tau_{j-1}, \tau_j)$ uniformly in $j \in \mathbb{N}$ for all $\tau_{j-1} \in \mathcal{T}$ since $A_{\mathrm{ref},s}$, $s \in \Sigma$, is Hurwitz, Σ is bounded, and $r(\cdot)$ is bounded. Furthermore, since Σ is bounded, it follows from (5.34) and (5.15) with $u(t) = \phi(\hat{\Theta}(t), \tilde{\Phi}_{\sigma(t)}(t))$ that $\dot{x}(\cdot)$ is bounded on $[\tau_{j-1}, \tau_j)$ uniformly in $j \in \mathbb{N}$ for all $\tau_{j-1} \in \mathcal{T}$. Therefore, $\dot{e}(\cdot)$ is bounded on $[\tau_{j-1}, \tau_j)$ uniformly in $j \in \mathbb{N}$ for all $\tau_{j-1} \in \mathcal{T}$ and $\ddot{V}(t, e(t), \hat{K}(t), \hat{K}_g(t)) = -2e^{\mathrm{T}}(t)Q\dot{e}(t)$ is bounded on $[\tau_{j-1}, \tau_j)$ uniformly in $j \in \mathbb{N}$ for all $\tau_{j-1} \in \mathcal{T}$. Consequently, it follows from Lemma 5.11 that $\dot{V}(t, e(t), \hat{K}(t), \hat{K}_g(t)) \to 0$ as $t \to \infty$ and hence, it follows from (5.37) that $e(t) \to 0$ as $t \to \infty$ uniformly in $t_0 \in [0, \infty)$ and $\sigma(\cdot)$, which concludes the proof. ■

Theorem 5.12 proves that if the matching conditions (5.17) and (5.18) are verified, the dwell-time t_{d} of the switching signal $\sigma(\cdot)$ is arbitrarily small, but non-zero, and there exists a solution to the Lyapunov inequality (5.28), then both the trajectory tracking error $e(\cdot)$ and the adaptive gain matrix $\hat{\Theta}(\cdot)$ are bounded and the trajectory of the closed-loop plant (5.15) with $u(t) = \phi(\hat{\Theta}(t), \tilde{\Phi}_{\sigma(t)})$, $t \geq t_0$, eventually mimics the trajectory of the reference model (5.16), that is, $\lim_{t\to\infty} \|e(t)\| = \lim_{t\to\infty} \|x(t) - x_{\mathrm{ref}}(t)\| = 0$ uniformly in both $t_0 \in [0, \infty)$ and $\sigma(\cdot)$.

5.3.3 Filippov framework

In this section, we address the model reference adaptive control design problem posed in Section 5.3.1 by analyzing Filippov solutions of the trajectory tracking error dynamics

(5.29) and the adaptive law (5.30) and assuming that the switching signal $\sigma(\cdot)$ is Lebesgue integrable and has countably many points of discontinuity over the time interval $[t_0, \infty)$. To this goal, define the *vectorized adaptive gain* $\hat{\theta}(t) \triangleq \mathrm{vec}(\hat{\Theta}(t))$, $t \geq t_0$, where $\mathrm{vec}(\cdot)$ denotes the vector-stacking operator, and consider the *vectorized adaptive law*

$$\dot{\hat{\theta}}(t) = -\mathrm{vec}\left(\Gamma \tilde{\Phi}_{\sigma(t)}(t, x(t)) e^{\mathrm{T}}(t) P B_{\sigma(t)}\right), \quad \hat{\theta}(t_0) = \mathrm{vec}(\hat{\Theta}_0), \quad t \geq t_0, \tag{5.38}$$

which has been deduced from (5.30). Furthermore, let $y(t) \triangleq \left[e^{\mathrm{T}}(t), \hat{\theta}^{\mathrm{T}}(t)\right]^{\mathrm{T}}$, $t \geq t_0$, and

$$f(t, y) \triangleq \begin{bmatrix} A_{\mathrm{ref},\sigma(t)} e + B_{\sigma(t)} \Delta \Theta^{\mathrm{T}} \tilde{\Phi}_{\sigma(t)}(t, x(t)) \\ -\mathrm{vec}\left(\Gamma \tilde{\Phi}_{\sigma(t)}(t, x(t)) e^{\mathrm{T}} P B_{\sigma(t)}\right) \end{bmatrix},$$
$$(t, y) \in [t_0, \infty) \times \mathbb{R}^{n+m(\overline{\sigma}(n+m)+N)}, \tag{5.39}$$

so that (5.29) and (5.30) are equivalent to

$$\dot{y}(t) = f(t, y(t)), \qquad y(t_0) = \begin{bmatrix} x_0 - x_{\mathrm{ref},0} \\ \mathrm{vec}(\hat{\Theta}_0) \end{bmatrix}, \qquad t \geq t_0. \tag{5.40}$$

It is worthwhile to note that the nonlinear, discontinuous dynamical system given by (5.40) is Lebesgue integrable and essentially locally bounded uniformly in $t \in [t_0, \infty)$ since $\Phi_s(\cdot, \cdot)$ is Lebesgue integrable, continuous in $t \in [t_0, \infty)$, and Lipschitz continuous in $x \in \mathcal{D}$, uniformly in t for all $s \in \Sigma$, and $\sigma(\cdot)$ is Lebesgue integrable and bounded.

Theorem 5.13. *Consider the closed-loop trajectory tracking error dynamics* (5.29) *and the adaptive law* (5.30). *Assume that the matching conditions* (5.17) *and* (5.18) *are verified and there exist symmetric positive-definite matrices* $P, Q \in \mathbb{R}^{n \times n}$ *so that* (5.35) *is verified. Then, every maximal solution of the Filippov regularization of* (5.29) *and* (5.30) *is bounded uniformly in both* $t_0 \in [0, \infty)$ *and* $\sigma(\cdot)$ *and such that* $e(t) \to 0$ *as* $t \to \infty$ *uniformly in both* t_0 *and* $\sigma(\cdot)$.

Proof: Consider the candidate common Lyapunov function

$$V(t,y) = e^{\mathrm{T}}Pe + \tilde{\theta}^{\mathrm{T}}\tilde{\theta}, \qquad (t,y) \in [t_0,\infty) \times \mathbb{R}^{n+m(\overline{\sigma}(n+m)+N)}, \tag{5.41}$$

where $\tilde{\theta} \triangleq \operatorname{vec}\left(\Gamma^{-\frac{1}{2}}\Delta\Theta\right)$, and note that if there exists $Q \in \mathbb{R}^{n\times n}$ that is symmetric, positive-definite, and such that (5.35) is verified, then the Lyapunov inequality (5.28) is verified. Since $V(\cdot,\cdot)$ is continuously differentiable, the Lyapunov function candidate (5.41) is Lipschitz continuous and regular, and it follows from Lemma 5.7 that $\dot{V}(t,y(t)) \in \dot{\overline{V}}(t,y(t))$, $t \in [t_0,\infty)$ a.e., where

$$\dot{\overline{V}}(t,y) \triangleq \bigcap_{\xi \in \partial V(t,y)} \xi^{\mathrm{T}} \begin{bmatrix} K[f](t,y) \\ 1 \end{bmatrix}, \qquad (t,y) \in [t_0,\infty) \times \mathbb{R}^{n+m(\overline{\sigma}(n+m)+N)}, \tag{5.42}$$

and $f(\cdot,\cdot)$ verifies (5.40). Furthermore, since $V(\cdot,\cdot)$ is continuously differentiable and does not depend on t explicitly, it holds that

$$\dot{\overline{V}}(t,y) \subset \frac{\partial V(t,y)}{\partial y} K\left[f\right](t,y) \subset 2\left[e^{\mathrm{T}}P, \tilde{\theta}^{\mathrm{T}}\right] K\left[f\right](t,y), \\ (t,y) \in [t_0,\infty) \times \mathbb{R}^{n+m(\overline{\sigma}(n+m)+N)}, \tag{5.43}$$

and since $f(t,\cdot)$ is continuous for all $t \in [t_0,\infty)$ and $f(\cdot,y)$ is continuous between switching times for all $y \in \mathbb{R}^{n+m(\overline{\sigma}(n+m)+N)}$, it follows from Theorem 1 of [166] that $K\left[f\right](t,y) = \{f(t,y)\}$ for all $(t,y) \in ([\tau_{j-1},\tau_j) \cap [t_0,t)) \times \mathbb{R}^{n+m(\overline{\sigma}(n+m)+N)}$, where $\tau_j \in \mathcal{T}$ and $j \in \mathbb{N}$. Therefore, by proceeding as in the proof of Theorem 5.12, it holds that

$$\dot{\overline{V}}(t,y(t)) \leq -\alpha_{\min}\|e(t)\|, \qquad t \in [t_0,\infty) \ \text{ a.e.}, \tag{5.44}$$

where $\alpha_{\min} = \lambda_{\min}(Q)$. Since $V(\cdot,\cdot)$ is positive-definite and radially unbounded and $W_3(y) = \alpha_{\min}\|e\|$ is a nonnegative-definite function of its argument, it follows from Theorem 5.8 that

every maximal solution $y(\cdot)$ of the Filippov regularization of (5.40) is bounded uniformly in both $t_0 \in [t_0, \infty)$ and $\sigma(\cdot)$ and such that $\lim_{t\to\infty} e(t) = 0$ uniformly in both t_0 and $\sigma(\cdot)$. ■

Theorem 5.13 proves that applying the control law (5.34) and the adaptive law (5.30) or, equivalently, (5.34) and (5.38), both the trajectory tracking error $e(\cdot)$ and the adaptive gain matrix $\hat{\Theta}(\cdot)$ are bounded, and $\lim_{t\to\infty} \|e(t)\| = 0$ uniformly in both $t_0 \in [0, \infty)$ and $\sigma(\cdot)$. Hence, it follows from the definition of limit that given $\varepsilon > 0$, there exists $T \geq t_0$ such that $\|e(t)\| < \varepsilon$ for $t \geq T$ a.e.. Theorem 5.12 proved a similar result, assuming that the dwell-time is non-zero. Theorem 5.13, instead, allows zero dwell-time. Moreover, Theorem 5.12 guarantees the existence of a unique solution of trajectory tracking error and the adaptive gains dynamics, whereas Theorem 5.13 guarantees the existence, but not the uniqueness, of a solution of (5.29) and (5.30).

5.4 Illustrative numerical example

In the this section, we provide a numerical example to demonstrate the effectiveness of both the control law (5.34) and the adaptive law (5.30) to guarantee that the trajectory $x(\cdot)$ of the plant (5.15) with $u(t) = \phi(\hat{\Theta}(t), \tilde{\Phi}_{\sigma(t)}(t))$, $t \geq t_0$, eventually tracks the trajectory $x_{\text{ref}}(\cdot)$ of the reference model (5.16). Specifically, we consider a delta-wing aircraft, whose wings' morphing mechanism is able to modify the vehicle's aerodynamic and geometric properties sufficiently fast to be considered as instantaneous, and whose aerodynamic and geometric coefficients are unknown. The roll dynamics of this vehicle is captured by

$$\begin{bmatrix} \dot{\varphi}(t) \\ \dot{p}(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \theta_{1,\sigma(t)} & \theta_{2,\sigma(t)} \end{bmatrix} \begin{bmatrix} \varphi(t) \\ p(t) \end{bmatrix} + \begin{bmatrix} 0 \\ \theta_{3,\sigma(t)} \end{bmatrix} \left[u(t) + \Theta^{\mathrm{T}} \Phi_{\sigma(t)}(t, \varphi(t), p(t)) \right],$$

$$\begin{bmatrix} \varphi(0) \\ p(0) \end{bmatrix} = \begin{bmatrix} \varphi_0 \\ p_0 \end{bmatrix}, \qquad t \geq 0, \tag{5.45}$$

where $\varphi(\cdot)$ denotes the *roll angle*, $p(\cdot)$ denotes the *roll rate*, $u(\cdot)$ denotes the *roll moment*, $\theta_{1,s}, \theta_{2,s}, \theta_{3,s} \in \mathbb{R}$ capture aerodynamic coefficients of the aircraft, $s \in \Sigma$, $\Sigma = \{1, 2\}$, $\theta_{1,s}$ and $\theta_{2,s}$ are unknown, $\Theta \in \mathbb{R}^4$ is unknown,

$$\Phi_s(t, \varphi, p) = \left[\mathbf{1}_{\{s \in \Sigma : s-1=0\}}(s)\bar{\Phi}_1(t, \varphi, p), \mathbf{1}_{\{s \in \Sigma : s-2=0\}}(s)\bar{\Phi}_2^{\mathrm{T}}(t, \varphi, p)\right]^{\mathrm{T}},$$
$$(s, t, \varphi, p) \in \Sigma \times [t_0, \infty) \times \mathbb{R} \times \mathbb{R},$$

denotes the regressor vector, and [75, pp. 285-291]

$$\bar{\Phi}_1(t, \varphi, p) = \tanh \varphi,$$
$$\bar{\Phi}_2(t, \varphi, p) = \left[|\varphi(t)|p(t), |p(t)|p(t), \varphi^3(t)\right]^{\mathrm{T}};$$

note that (5.45) is in the same form as (5.15) with $n = 2$, $m = 1$, $t_0 = 0$, $x = [\varphi, p]^{\mathrm{T}}$, $A_s = \begin{bmatrix} 0 & 1 \\ \theta_{1,s} & \theta_{2,s} \end{bmatrix}$, $s \in \Sigma$, $\overline{\sigma} = 2$, and $B_s = \begin{bmatrix} 0 \\ \theta_{3,s} \end{bmatrix}$. Additionally, we consider the switched reference model (5.16) with $x_{\mathrm{ref}}(t) = [\varphi_{\mathrm{ref}}(t), p_{\mathrm{ref}}(t)]^{\mathrm{T}}$, $A_{\mathrm{ref},s} = \begin{bmatrix} 0 & 1 \\ -k_s & -c_s \end{bmatrix}$, $s \in \Sigma$, $k_s > 0$, $c_s > 0$, $B_{\mathrm{ref,s}} = \begin{bmatrix} 0 \\ b_s \end{bmatrix}$, and $b_s \in \mathbb{R}$ so that if $\sigma(t) = 1$, $t \geq 0$, then (5.16) captures a less responsive reference model, and if $\sigma(t) = 2$, then (5.16) captures a more responsive reference model. It is worthwhile to note that $A_{\mathrm{ref},s}$, $s \in \Sigma$, is in companion form and hence, there exists a symmetric positive-definite matrix $P \in \mathbb{R}^{2\times 2}$ that verifies (5.28) if and only if the matrix product $A_{\mathrm{ref},1}A_{\mathrm{ref},2}$ does not have any negative real eigenvalue [193].

Let $\theta_{1,1} = -9.15$, $\theta_{2,1} = -4.6$, $\theta_{3,1} = 1$, $\theta_{1,2} = -0.018$, $\theta_{2,2} = 0.015$, $\theta_{3,2} = 0.75$, $\Theta = [1, -0.062, 1, 0.009]^{\mathrm{T}}$, $k_1 = 1$, $c_1 = 3$, $b_1 = 1$, $k_2 = 150$, $c_2 = 45$, $b_2 = 150$, $r(t) = \frac{1}{2}\sin 2t$, $t \geq 0$, $\sigma(t) = \mathrm{rpi}\left(\frac{1}{2}\sin(\omega(t)t) + \frac{3}{2}\right)$, where $\mathrm{rpi}(\cdot)$ denotes the rounding function to the nearest

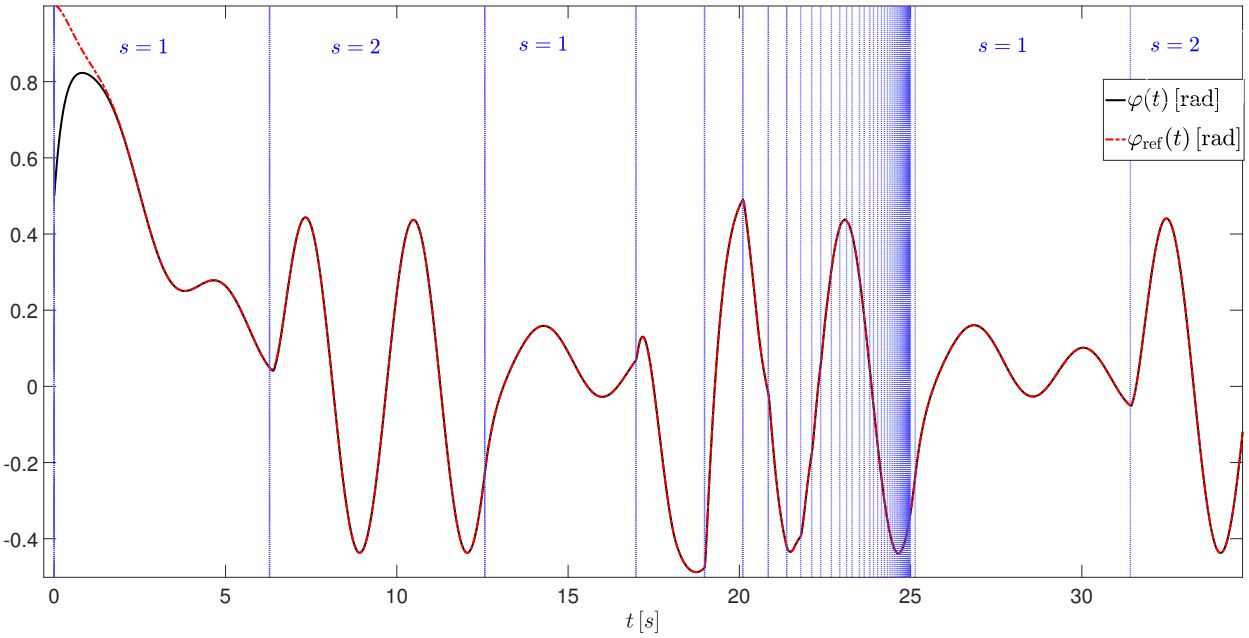

Figure 5.1: Plot of the aircraft's roll angle $\varphi(\cdot)$ and reference roll angle $\varphi_{\text{ref}}(\cdot)$. The vertical lines mark the switching times. Despite the arbitrarily small dwell-time and the uncertainties in both the plant dynamics and the initial conditions, the aircraft's roll angle tracks closely the reference roll angle.

integer,

$$\omega(t) = \begin{cases} 0.5, & t \in [0, 15) \cup [25, \infty), \\ \dfrac{3(t-15)}{4(25-t)}, & t \in [15, 25), \end{cases}$$

$\Gamma = 50 I_{10}$, $\varphi_0 = 1$, and $p_0 = 1$; note that the dwell-time converges to zero as $t \to 25$ from the left. In this case, $\text{spec}(A_1) = \{-2.3000 - 1.9647\jmath, -2.3000 + 1.9647\jmath\}$ and $\text{spec}(A_2) = \{0.0075 - 0.1340\jmath, 0.0075 + 0.1340\jmath, \}$, which implies that the uncontrolled plant is asymptotically stable for $s = 1$ and is unstable for $s = 2$. Furthermore, $\text{spec}(A_{\text{ref},1} A_{\text{ref},2}) = \{-8.000 - 9.2736\jmath, -8.000 + 9.2736\jmath\}$, and (5.28) is verified by $P = \begin{bmatrix} 2.99 & 0.75 \\ 0.75 & 0.288 \end{bmatrix} \cdot 10^4$.

Figure 5.1 shows plots of the aircraft's roll angle and the corresponding reference trajectory. The control law (5.34) and the adaptive law (5.30) guarantee satisfactory trajectory tracking despite uncertainties in the plant dynamics and the initial conditions and despite

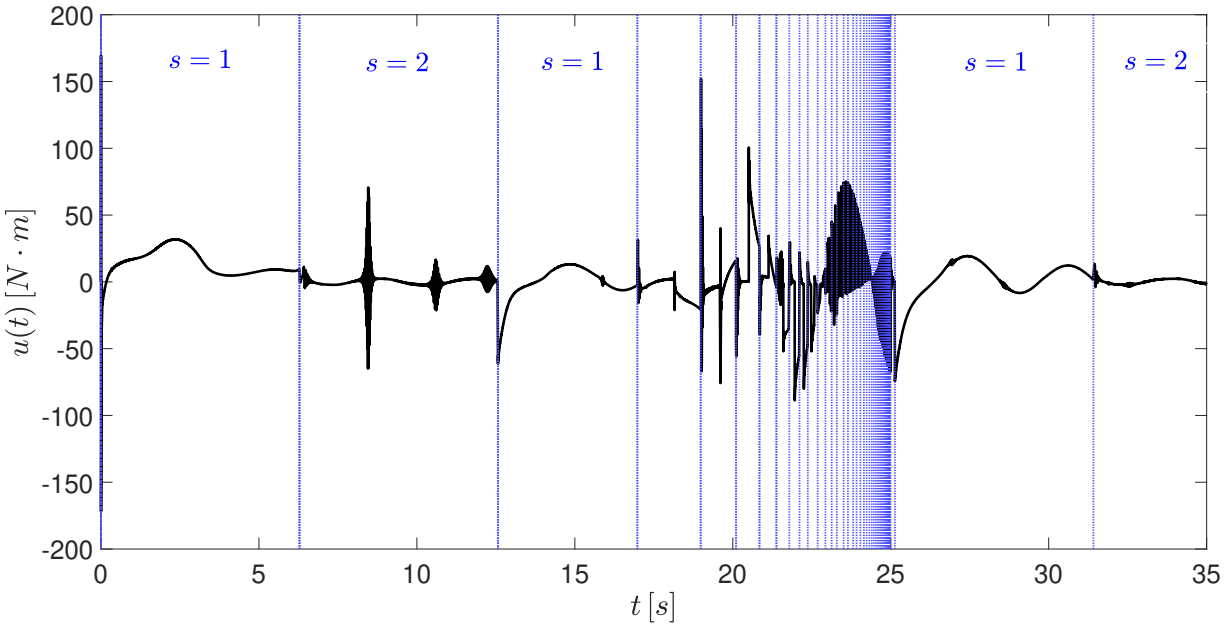

Figure 5.2: Plot of control input obtained applying the control law (5.34) and the proposed adaptive law (5.30). Both the magnitude an the frequency of the control input increase with the frequency of the switching signal.

the arbitrarily small dwell-time over the interval $[15, 25)$ s. Figure 5.2 shows a plot of the roll moment needed to track the reference trajectory. It is apparent that each switching is followed by a sudden increase of the control input, and over the time interval $[15, 25)$ s, the control input is characterized by high-frequency oscillations due to the rapid sequence of switching times.

None of the control techniques for uncertain, switched, nonlinear plants that we surveyed is suitable to regulate (5.45). Therefore, to validate the usefulness of the adaptive law (5.30), we considered the problem of applying the control law (5.34) and the classical adaptive law

$$\dot{\hat{\Theta}}(t) = -\Gamma \tilde{\Phi}_s(t, x(t)) e^{\mathrm{T}}(t) P B_s, \qquad s \in \Sigma, \qquad \hat{\Theta}(t_0) = \hat{\Theta}_0, \qquad t \geq t_0, \tag{5.46}$$

to regulate the aircraft's roll dynamics. If $s = 1$ in both (5.34) and (5.46), then the trajectory tracking error diverges. Alternatively, if $s = 2$ in both (5.34) and (5.46), then the aircraft is able to track the reference roll angle. However, as shown in Figure 5.3, applying (5.34)

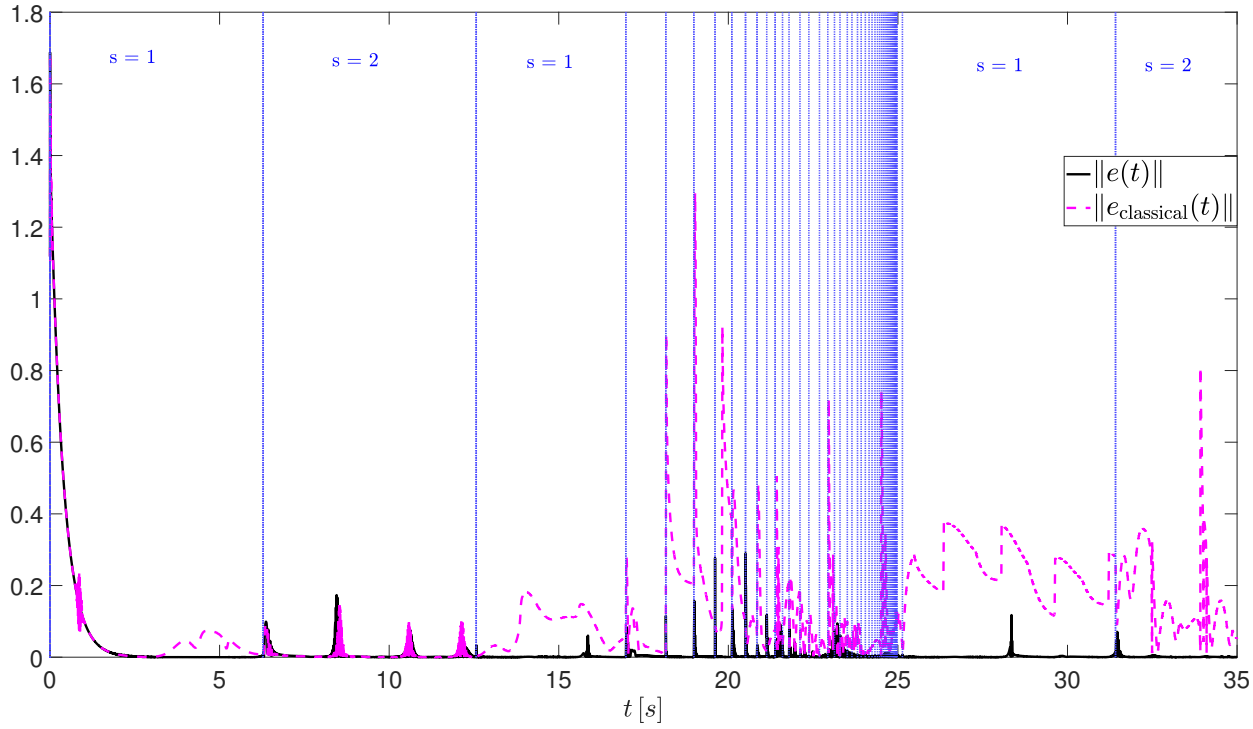

Figure 5.3: Plot of the trajectory tracking error norm obtained by applying the control law (5.34) and the proposed adaptive law (5.30), namely $\|e(\cdot)\|$, and plot of the trajectory tracking error norm obtained by applying the control law (5.34) and the classical adaptive law (5.46) with $s = 2$, namely $\|e_{\text{classical}}(\cdot)\|$. Applying (5.34) and (5.30), the trajectory tracking error is consistently smaller.

and (5.46) with $s = 2$, the trajectory tracking error is consistently larger than the trajectory tracking error obtained by applying the proposed framework. Indeed, applying (5.34) and (5.30), the $\mathcal{L}_2$-norm of the trajectory tracking error is equal to 0.049N, and applying (5.34) and (5.46) with $s = 2$, the $\mathcal{L}_2$-norm of the trajectory tracking error is equal to 0.209N. Figure 5.4 shows the control input obtained applying (5.34) and (5.46) and $s = 2$. By comparing Figures 5.2 and 5.4, it appears that applying (5.34) and (5.46) with $s = 2$, the control effort is three orders of magnitude larger than the control effort needed to apply (5.34) and (5.30); indeed, the $\mathcal{L}_\infty$-norm of the control input obtained by applying (5.34) and (5.46) with $s = 2$ is $415,559.07$N, whereas the $\mathcal{L}_\infty$-norm of the control input obtained applying (5.34) with the proposed adaptive law (5.30) is 171.75N. Lastly, we remark that, employing (5.34) and (5.30), the computational time is approximately 10.4 times shorter than employing (5.34)

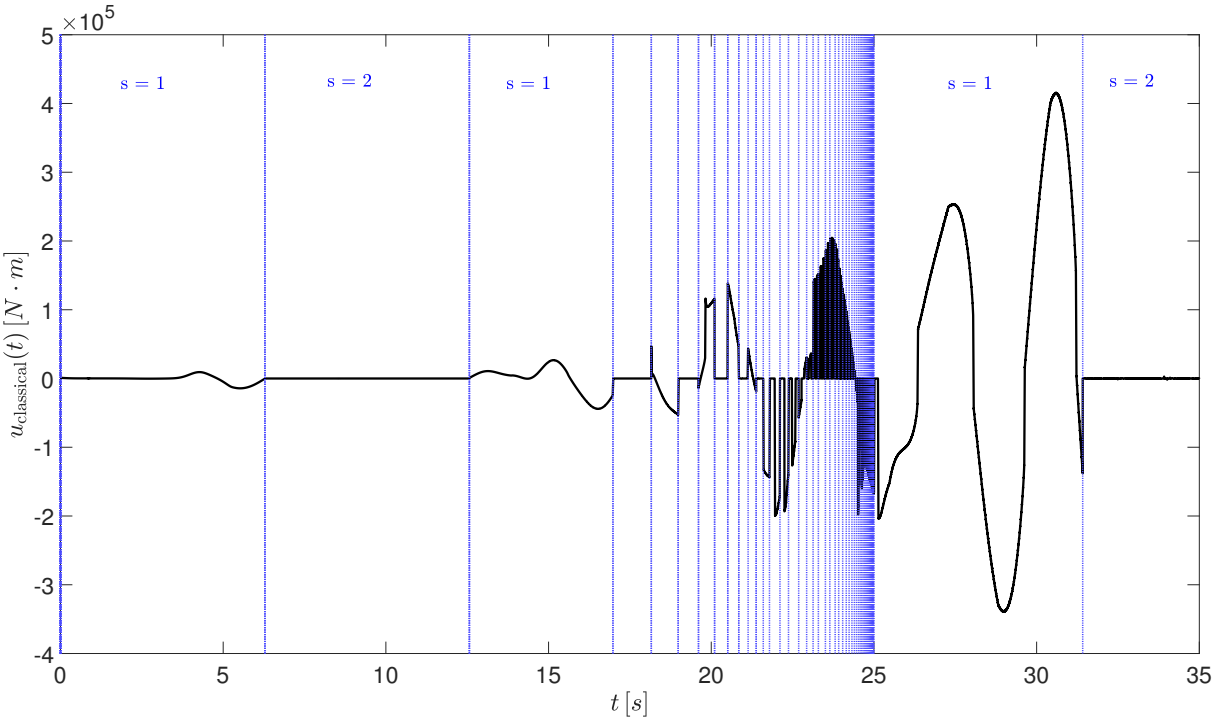

Figure 5.4: Plot of control input obtained applying the control law (5.34) and the classical adaptive law (5.46) with $s = 2$. By comparing this plot with the plot in Figure 5.2, it is apparent that applying (5.34) and (5.46) with $s = 2$, the control effort is three orders of magnitude larger than the control effort needed to apply (5.34) and (5.30).

and (5.46) with $s = 2$.

Since transient performance can be a known issue with MRAC in general, addressed in Chapter 4, an additional simulation is performed where the switching becomes arbitrarily fast at 1s simulation time during the transient. The switching law is given by,

$$\omega(t) = \begin{cases} \dfrac{3(t-1)}{4(1-t)}, & t \in [0, 1), \\ 0.25(t-1), & t \in [1, 10). \end{cases}$$

As seen in Figure 5.5, the system still converges to the reference signal in similar fashion as the previous simulation.

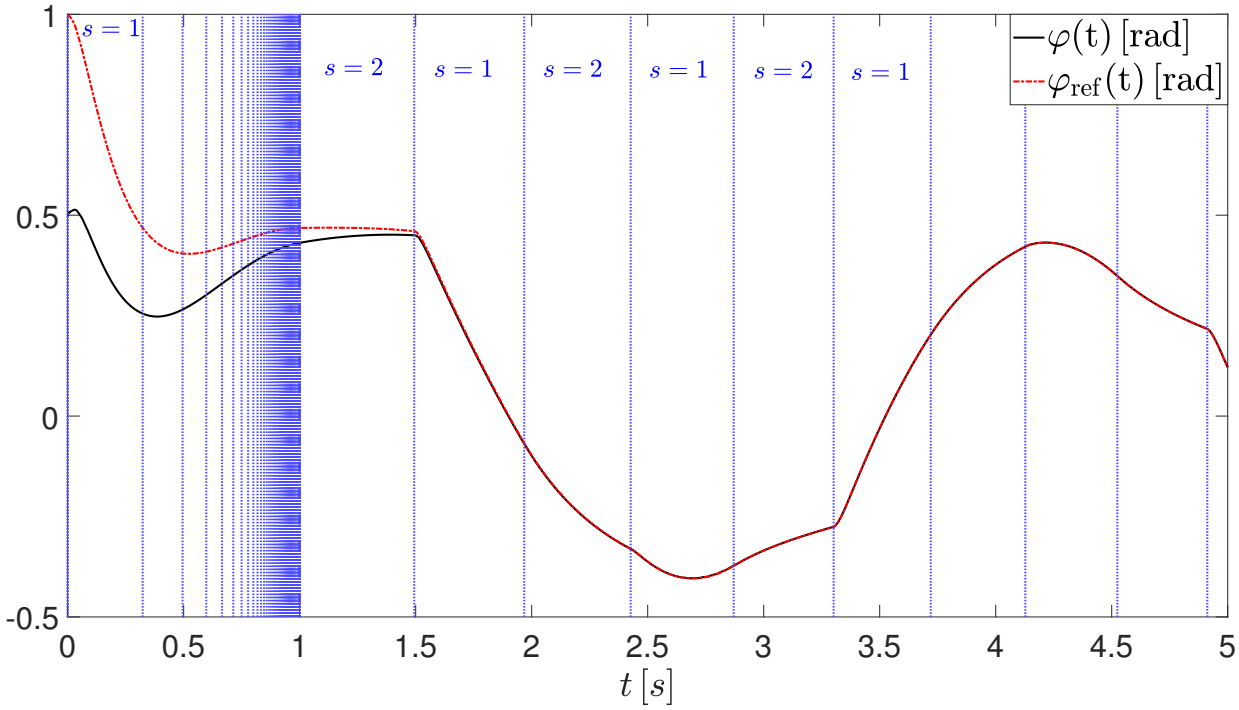

Figure 5.5: Plot of the aircraft's roll angle $\varphi(\cdot)$ and reference roll angle $\varphi_{\text{ref}}(\cdot)$. The vertical lines mark the switching times. Despite the arbitrarily small dwell-time during the transient and the uncertainties in both the plant dynamics and the initial conditions, the aircraft's roll angle tracks closely the reference roll angle.

5.5 Flight tests

5.5.1 Problem description

In order to validate the proposed model reference adaptive control framework for unknown switched nonlinear plants, we performed flight tests involving a tilt-rotor UAV tasked with autonomously mounting a camera to a vertical surface. This UAV comprises a chassis, which is modeled as a rigid body, four propellers, whose spin axes can be tilted independently, a robotic arm, and a grasper, which comprises a suction cup mounted at the extremity of the robotic arm; for details, see Figure 5.6. The camera is held by the grasper and is covered by one of the two sides of a lineal fabric strip fastener. The point on the vertical surface where the camera must be installed is covered by the other side of the fabric strip fastener. After having exerted some normal force against the vertical surface and having engaged the fabric

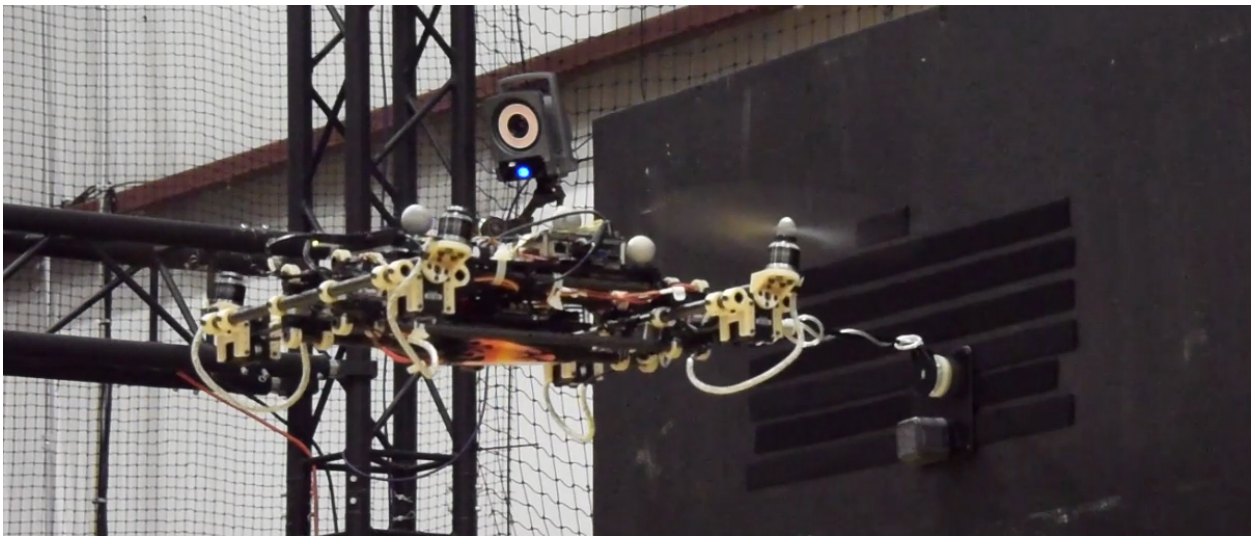

Figure 5.6: Tilt-rotor UAV installing a camera on a vertical surface. The camera is held by the grasper and is covered by one of the two sides of a lineal fabric strip fastener. The vertical surface is partly covered by the other side of the fabric strip fastener so that, after having exerted some pressure against the vertical surface, the suction cup is released, and the UAV flies away.

strip fastener, the suction cup is released and the UAV flies away from the vertical surface. A video of one of these flight tests can be found at [194].

5.5.2 Dynamical modeling

To uniquely identify the position and orientation of the UAV in space, we consider two reference frames, namely the *inertial reference frame* $\mathbb{I} \triangleq \{O; X, Y, Z\}$ centered in $O \in \mathbb{R}^3$ and with orthonormal axes $X, Y, Z \in \mathbb{R}^3$ and the *body reference frame* $\mathbb{J}(\cdot) \triangleq \{A(\cdot); x(\cdot), y(\cdot), z(\cdot)\}$ centered at the extremity of the arm $A : [t_0, \infty) \to \mathbb{R}^3$ and with orthonormal axes $x, y, z : [t_0, \infty) \to \mathbb{R}^3$. If a vector $a \in \mathbb{R}^3$ is expressed in the reference frame $\mathbb{I}$, then it is denoted by $a^{\mathbb{I}}$; alternatively, if a vector is expressed in $\mathbb{J}(\cdot)$, then no superscript is used. The reference frame $\mathbb{I}$ is set so that the X axis is normal to the vertical surface and the force due to the gravitational acceleration is given by $F_{\mathrm{g}}^{\mathbb{I}} = -mgZ$, where $m > 0$ denotes the mass of the UAV and $g > 0$ denotes the gravitational acceleration. The reference frame $\mathbb{J}(\cdot)$ is set so that the propellers' arms are aligned to the $y(\cdot)$ axis; for details, see Figure 5.7.

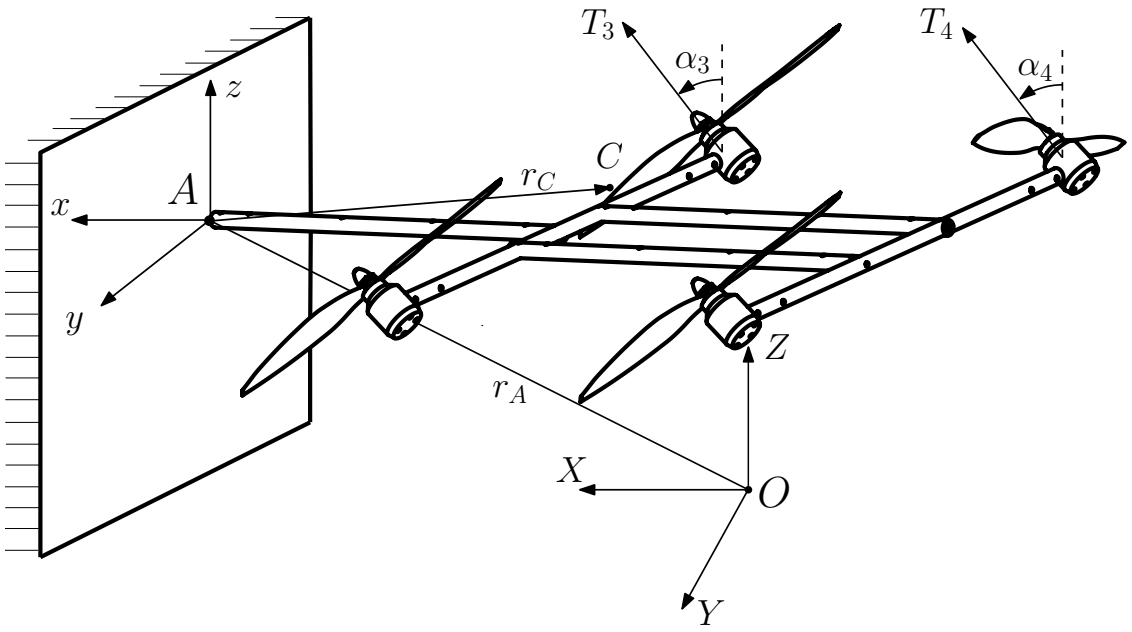

Figure 5.7: Schematic representation of a simplified unmanned aerial manipulator system exerting a normal force against a vertical surface.

The position of the reference point $A(\cdot)$ with respect to O is denoted by $r_A^{\mathbb{I}} : [t_0, \infty) \to \mathbb{R}^3$ and the *velocity* of $A(\cdot)$ with respect to the reference frame $\mathbb{I}$ is denoted by $v_A^{\mathbb{I}} : [t_0, \infty) \to \mathbb{R}^3$. Using a 3-2-1 rotation sequence, the orientation of the body reference frame $\mathbb{J}(\cdot)$ with respect to the inertial reference frame $\mathbb{I}$ is captured by the *roll angle* $\phi : [t_0, \infty) \to [0, 2\pi)$, the *pitch angle* $\theta : [t_0, \infty) \to \left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$, and the *yaw angle* $\psi : [t_0, \infty) \to [0, 2\pi)$ [125, pp. 11].

The *vector of independent generalized coordinates*

$$q(t) \triangleq \left[\left(r_A^{\mathbb{I}}(t)\right)^{\mathrm{T}}, \phi(t), \theta(t), \psi(t)\right]^{\mathrm{T}} \in \mathcal{D}, \qquad t \geq t_0, \tag{5.47}$$

captures the position and orientation of $\mathbb{J}(\cdot)$ with respect to $\mathbb{I}$, where $\mathcal{D} \triangleq \mathbb{R}^3 \times \left(-\frac{\pi}{2}, \frac{\pi}{2}\right) \times \left(-\frac{\pi}{2}, \frac{\pi}{2}\right) \times [0, 2\pi)$. The *kinematic equations* of the UAV are given by

$$\dot{q}(t) = \begin{bmatrix} v_A^{\mathbb{I}}(t) \\ \Gamma(q(t))\omega(q(t), \dot{q}(t)) \end{bmatrix}, \qquad q(t_0) = q_0, \qquad t \geq t_0, \tag{5.48}$$

where $\omega : \mathcal{D} \times \mathbb{R}^6 \to \mathbb{R}^3$ denotes the *angular velocity* of the reference frame $\mathbb{J}(\cdot)$ with respect

to $\mathbb{I}$, and [125, Th. 1.7]

$$\Gamma(q) \triangleq \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix}, \qquad q \in \mathcal{D};$$

it is worthwhile to recall that $\Gamma(q)$ is invertible, since $\theta \in (-\frac{\pi}{2}, \frac{\pi}{2})$ [125, pp. 18-19].

After the fabric strip fastener has been engaged and before the suction cup is released, the point of contact between the robotic arm and the vertical surface is modeled as a cylindrical hinge since the aircraft can only rotate about the $y(\cdot)$ axis. Furthermore, the translation of the reference point $A(\cdot)$, the aircraft's roll angle, and the aircraft's yaw angle are impeded. Therefore, while the UAV is in contact with the vertical surface, the plant dynamics is in partial-state equilibrium [117, Def. 4.1]. Furthermore, the UAV's dynamics comprises two models, that is, $\Sigma = \{1, 2\}$. Specifically, the first model captures the UAV's free flight dynamics, and the state vector comprises twelve components, that is, the components of $[q^{\mathrm{T}}(\cdot), \dot{q}^{\mathrm{T}}(\cdot)]^{\mathrm{T}}$. The second dynamical model captures the UAV's pitch dynamics while in contact with the vertical surface, the state vector comprises two components, that is, $[\theta(\cdot), \dot{\theta}(\cdot)]^{\mathrm{T}}$, and the remaining components of $[q^{\mathrm{T}}(\cdot), \dot{q}^{\mathrm{T}}(\cdot)]^{\mathrm{T}}$ are at equilibrium. By proceeding as in [20], the translational and rotational dynamic equations of the UAV are given by

$$\mathcal{H}_{\sigma(t)}\mathcal{M}(q(t)) \begin{bmatrix} \dot{v}_A^{\mathbb{I}}(t) \\ \dot{\omega}(q(t), \dot{q}(t)) \end{bmatrix} = \mathcal{H}_{\sigma(t)} \left(\begin{bmatrix} f_{\mathrm{dyn,tran}}(t, q(t), \dot{q}(t)) \\ f_{\mathrm{dyn,rot}}(t, q(t), \dot{q}(t)) \end{bmatrix} + G(q(t))u(t) \right),$$

$$\left[v_A^{\mathbb{I},\mathrm{T}}(t_0), \omega^{\mathrm{T}}(t_0) \right]^{\mathrm{T}} = \left[v_{A,0}^{\mathbb{I},\mathrm{T}}, \omega_0^{\mathrm{T}} \right]^{\mathrm{T}}, \qquad t \geq t_0, \tag{5.49}$$

where

$$\mathcal{H}_s \triangleq \begin{bmatrix} \mathbf{1}_{\{s\in\Sigma:s-1=0\}}(s) I_3 & 0_{3\times 3} \\ 0_{3\times 3} & \mathcal{I}_{\rm rot}(s) \end{bmatrix}, \qquad s \in \Sigma, \tag{5.50}$$

$\mathcal{I}_{\rm rot}(s) \triangleq \operatorname{diag}\left(\mathbf{1}_{\{s\in\Sigma:s-1=0\}}(s), 1, \mathbf{1}_{\{s\in\Sigma:s-1=0\}}(s)\right)$,

$$\mathcal{M}(q) \triangleq \begin{bmatrix} mI_3 & -mR(q)r_C^{\times} \\ mr_C^{\times}R^{\rm T}(q) & \mathfrak{I} \end{bmatrix}, \qquad q \in \mathcal{D}, \tag{5.51}$$

denotes the *generalized mass matrix*, $r_C \in \mathbb{R}^3$ denotes the position of center of mass of the controlled mechanical system with respect to the reference point $A(\cdot)$, the symmetric, positive-definite matrix $\mathfrak{I} \in \mathbb{R}^3$ denotes the *inertia matrix of the UAV* with respect to the reference point $A(\cdot)$,

$$f_{\rm dyn,tran}(t,q,\dot{q}) \triangleq F_g^{\mathbb{I}} - mR(q)\omega^{\times}(q,\dot{q})\omega^{\times}(q,\dot{q})r_C, \tag{5.52}$$

$$\begin{aligned} f_{\rm dyn,rot}(t,q,\dot{q}) \triangleq & -\omega^{\times}(q,\dot{q})\mathfrak{I}\omega(q,\dot{q}) - \sum_{i=1}^{4}\left[\mathfrak{I}_{P_i}(t)\dot{\omega}_{P_i}(t) + \omega_{P_i}^{\times}(t)\mathfrak{I}_{P_i}(t)\omega_{P_i}(t)\right] \\ & - \omega^{\times}(q,\dot{q})\sum_{i=1}^{4}\mathfrak{I}_{P_i}(t)\omega_{P_i}(t) + r_C^{\times}R^{\rm T}(q)F_{\rm g}^{\mathbb{I}}, \end{aligned} \tag{5.53}$$

$$G(q) \triangleq \begin{bmatrix} R(q)\begin{bmatrix} \mathbf{e}_{1,3} & \mathbf{e}_{3,3} \end{bmatrix} & 0_{3\times 3} \\ 0_{3\times 2} & I_3 \end{bmatrix}, \tag{5.54}$$

$u \triangleq [u_5, u_1, \ldots, u_4]^{\rm T} \in \mathbb{R}^5$ denotes the control input, $u_5, u_1 : [t_0, \infty) \to \mathbb{R}$, denote the *components of the forces produced by the propellers along the $x(\cdot)$ and $z(\cdot)$ axes*, respectively, and $[u_2, u_3, u_4]^{\rm T} : [t_0, \infty) \to \mathbb{R}^3$ denotes the *moment of the force produced by the propellers*. The inertia matrix of the ith propeller $\mathfrak{I}_{P_i}(\cdot)$, $i = 1, \ldots, 4$, is a function of time since the propellers' tilt angle may vary [20]. The first component of the control vector $u(\cdot)$ is denoted by $u_5(\cdot)$ for consistency with the notation concerning classical quadcopters, for which u :

$[t_0, \infty) \to \mathbb{R}^4$ and $u_5(t) \equiv 0$, $t \geq t_0$; for details, see [74].

If $\sigma(t) = 1$, $t \in \bigcup_{k=0}^{\infty}[\tau_{2k}, \tau_{2k+1})$, then $\mathcal{H}_{\sigma(t)} = I_6$, and (5.49) captures the aircraft's free flight dynamics. Alternatively, if $\sigma(t) = 2$, $t \in \bigcup_{j=0}^{\infty}[\tau_{2j+1}, \tau_{2(j+1)})$, then $\mathcal{H}_{\sigma(t)} = \begin{bmatrix} 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \end{bmatrix}$, and (5.49) captures the pitch dynamics of the UAV while connected to the vertical surface. An equivalent formulation of the UAV's dynamics may have been deduced by setting $\mathcal{H}_{\sigma(t)} = I_6$, $t \geq t_0$, and introducing the impulsive reaction forces and moments imposed by the vertical surface in the UAV's free flight dynamic equations.

5.5.3 Control strategy

Let $q_{\mathrm{ref}}(t) \triangleq \left[\left(r^{\mathbb{I}}_{\mathrm{ref}}(t)\right)^{\mathrm{T}}, \phi_{\mathrm{ref}}(t), \theta_{\mathrm{ref}}(t), \psi_{\mathrm{ref}}(t)\right]^{\mathrm{T}} \in \mathcal{D}$, $t \geq t_0$, denote the piece-wise twice continuously differentiable *reference vector of independent generalized coordinates*, where $r^{\mathbb{I}}_{\mathrm{ref}}(\cdot)$ captures the user-defined *reference trajectory* for the point $A(\cdot)$, $\phi_{\mathrm{ref}}(\cdot)$ captures the *reference roll angle*, $\theta_{\mathrm{ref}}(\cdot)$ captures the user-defined *reference pitch angle*, and $\psi_{\mathrm{ref}}(\cdot)$ captures the user-defined *reference yaw angle*. We design $q_{\mathrm{ref}}(\cdot)$ so that $r^{\mathbb{I}}_{\mathrm{ref}}(t) = r^{\mathbb{I}}_{\mathrm{wall}}$, $t \in \bigcup_{j=0}^{\infty}[\tau_{2j+1}, \tau_{2(j+1)})$, where $r^{\mathbb{I}}_{\mathrm{wall}}$ denotes the point on the vertical surface where the camera must be installed. Furthermore, we set $\theta_{\mathrm{ref}}(t) \equiv 0$, $t \geq t_0$, and $\psi_{\mathrm{ref}}(t) \equiv 0$ so that the robotic arm is orthogonal to the vertical surface upon impact. The UAV considered in this research is underactuated since it is characterized by six degrees of freedom, namely the components of $q(\cdot)$, and five control inputs, namely the components of $u(\cdot)$. Therefore, it is impossible to define $q_{\mathrm{ref}}(\cdot)$ arbitrarily. For these aerial vehicles, $\phi_{\mathrm{ref}}(\cdot)$ is deduced so that the desired displacement of the reference point $A(\cdot)$ along the direction Y of the reference frame $\mathbb{I}$ can be achieved [20]. In this chapter, we set $\mathbf{e}^{\mathrm{T}}_{2,3} r^{\mathbb{I}}_{\mathrm{ref}}(t) = 0$, $t \geq t_0$, so that if $\left|\mathbf{e}^{\mathrm{T}}_{2,3}\left(r^{\mathbb{I}}_{A}(t) - r^{\mathbb{I}}_{\mathrm{ref}}(t)\right)\right| = 0$, $t \geq t_0$, then $\phi_{\mathrm{ref}}(t) = 0$.

Next, consider the feedback-linearizing control law

$$\beta_s(t,q,q_{\rm ref},w) \triangleq h\Bigg(- \begin{bmatrix} f_{\rm dyn,tran}(t,q,\dot{q}) \\ f_{\rm dyn,rot}(t,q,\dot{q}) \end{bmatrix} + \mathcal{M}(q) \begin{bmatrix} \mathbf{1}_3 & 0_{3\times 3} \\ 0_{3\times 3} & \Gamma^{-1}(q) \end{bmatrix} \cdot \left[\ddot{q}_{\rm ref} - \begin{bmatrix} 0_{3\times 1} \\ \dot{\Gamma}(q)\omega(q,\dot{q}) \end{bmatrix} - [K_{{\rm P},s}, K_{{\rm D},s}] \begin{bmatrix} q - q_{\rm ref} \\ \dot{q} - \dot{q}_{\rm ref} \end{bmatrix} + w \right] \Bigg),$$

$$(s,t,q,q_{\rm ref},w) \in \Sigma \times [t_0,\infty) \times \mathcal{D} \times \mathcal{D} \times \mathbb{R}^6, \tag{5.55}$$

where $K_{{\rm P},s}, K_{{\rm D},s} \in \mathbb{R}^6$ are user-defined, symmetric, and positive-definite gain matrices that define a proportional-derivative baseline controller, $h : \mathbb{R}^n \to \mathbb{R}^n$ is defined so that $\mathbf{e}_{j,6}^{\rm T} h(x) = x_j$, $j \in \{1,2,4,5,6\}$, and $\mathbf{e}_{3,6}^{\rm T} h(x) = \mu_\kappa(x_3)$, and $\mu_\kappa : \mathbb{R} \to \mathbb{R}$ is defined so that $\mu_\kappa(\alpha) = \kappa \, {\rm sign}\alpha$, for $|\alpha| \leq \kappa$ and $\kappa > 0$ user-defined and arbitrarily small, and $\mu_\kappa(\alpha) = \alpha$, for $|\alpha| > \kappa$. If $u(t) = \beta_{\sigma(t)}(t, q(t), q_{\rm ref}(t), w(t))$, $t \geq t_0$, then the trajectory tracking error dynamics is given by the switched dynamical model

$$\dot{e}(t) = A_{{\rm ref},\sigma(t)} e(t) + B \left[w(t) + \tilde{\Theta}^{\rm T} \tilde{\Phi}_{\sigma(t)}(t, q(t), \dot{q}(t)) \right],$$

$$e(t_0) = \begin{bmatrix} q(t_0) - q_{\rm ref}(t_0) \\ \dot{q}(t_0) - \dot{q}_{\rm ref}(t_0) \end{bmatrix}, \qquad t \geq t_0, \tag{5.56}$$

where $e(t) \triangleq q(t) - q_{\rm ref}(t)$, denotes the *trajectory tracking error*, $A_{{\rm ref},s} = \begin{bmatrix} 0_{6\times 6} & I_6 \\ -K_{{\rm P},s} & -K_{{\rm D},s} \end{bmatrix}$, $s \in \Sigma$, $B = \begin{bmatrix} 0_{6\times 6} \\ I_6 \end{bmatrix}$, $\tilde{\Theta} \in \mathbb{R}^{39\times 6}$ is unknown, and

$$\tilde{\Phi}_s(t,q,\dot{q}) \triangleq \left[\mathbf{1}_{\{s\in\Sigma : s-1=0\}}(s) \overline{\Phi}_1^{\rm T}(t,q,\dot{q}), \mathbf{1}_{\{s\in\Sigma : s-2=0\}}(s) \overline{\Phi}_2(t,q,\dot{q}), \right]^{\rm T},$$

$$(s,t,q,\dot{q}) \in \Sigma \times [t_0,\infty) \times \mathcal{D} \times \mathbb{R}^6, \tag{5.57}$$

$$\overline{\Phi}_1(t,q,\dot{q}) = \big[-m W_M^{\rm T} (R(q)\omega^\times(q,\dot{q})\omega^\times(q,\dot{q})),$$

$$-W_M^{\mathrm{T}}\left(\omega^{\times}(q,\dot{q})M_W(\omega(q,\dot{q}))\right),F_g^{\mathrm{T}}(q)\big]^{\mathrm{T}}, \tag{5.58}$$

$$\overline{\Phi}_2(t,q,\dot{q}) = \mathbf{e}_{3,3}^{\mathrm{T}}F_g(q), \tag{5.59}$$

$$W_M(A) \triangleq \sum_{i=1}^{n}[\mathbf{e}_{i,m}\otimes(A\mathbf{e}_{i,m})], \qquad A\in\mathbb{R}^{n\times m}, \tag{5.60}$$

$$M_W(b,n) \triangleq \left(b^{\mathrm{T}}\otimes I_n\right), \qquad (b,n)\in\mathbb{R}^m\times\mathbb{N}, \qquad b\in\mathbb{R}^n. \tag{5.61}$$

The regressor vector $\tilde{\Phi}_s(t,q,\dot{q})$, $(s,t,q,\dot{q})\in\Sigma\times[t_0,\infty)\times\mathcal{D}\times\mathbb{R}^6$, has been constructed to capture the effect of uncertainties on the location of the UAV's center of mass on the right-hand side of the dynamic equation (5.49). The function $h(\cdot)$ in (5.55) guarantees the controllability of the closed-loop plant dynamics at all times [20].

Let $w(t)=\phi(\hat{\Theta}(t),\tilde{\Phi}_{\sigma(t)}(t,q(t),\dot{q}(t)))$, $t\geq t_0$, where the feedback control law $\phi(\cdot,\cdot)$ is given by (5.34) and $\hat{\Theta}(\cdot)$ verifies the adaptive law (5.30). In this case, the trajectory tracking error dynamics (5.56) is in the same form as (5.29) with $n=12$, $m=6$, $x=\left[q^{\mathrm{T}},\dot{q}^{\mathrm{T}}\right]^{\mathrm{T}}$, and $B_s=B$, $s\in\Sigma$. Therefore, it follows from Theorem 5.12 or, alternatively, from Theorem 5.13 that the proposed model reference adaptive control law (5.34) and the proposed adaptive law (5.30) can be employed to guarantee satisfactory trajectory tracking at all times.

5.5.4 Flight tests setup

The UAV employed to test the proposed model reference adaptive control framework for switched dynamical systems is custom designed. The chassis is made of carbon fiber rods and the propellers are actuated by Dynamixel AX-18A servo motors, which guarantee ±1 deg precision and allow to measure their displacement. To guarantee successful grasping and placement of the camera sensor and allow the user to release the camera at a desired time instant, a gripper based on an active, self-sealing suction cup has been employed.

The proposed control law has been coded in the C++ programming language and implemented on an ODroid XU4 companion computer, which integrates both (5.30) and (5.29)

and evaluates the control law (5.34) at a frequency of approximately 250 Hz. Once the control input $u(\cdot)$ has been determined, the companion computer calculates the tilt angle and thrust force for each propeller to realize the desired control input according to the algorithm outlined in [20]. The servo actuators that regulate the propellers' tilt angles are controlled directly by the companion computer. Each propeller's desired thrust force is transmitted from the companion computer to a Pixhawk 2 flight controller over a dedicated serial line, and the flight controller coordinates each propeller so that the desired thrust force is realized. The flight controller also embeds an inertial measurement unit and an extended Kalman filter to estimate the vehicle's rotational position and velocities. Flight tests have been performed indoors, and the UAV's position and velocity are deduced by a Vicon motion capture system and transmitted over WiFi to the companion computer.

5.5.5 Flight tests results

The feedback-linearizing control law (5.55) and the adaptive law (5.30) have been coded assuming that the UAV's mass is $m = 2.06$kg, which has been deduced using a precision scale, and its inertia matrix is

$$\mathfrak{I} = \begin{bmatrix} 1.97 & -0.0732 & -0.0182 \\ -0.0732 & 1.37 & -0.0162 \\ -0.0182 & -0.0162 & 10.8 \end{bmatrix} \cdot 10^{-3} kg \cdot m^2,$$

which has been deduced using a computer aided design (CAD) model. Furthermore, the ith propeller's inertia matrix $\mathfrak{I}_{P_i}(\cdot)$, $i = 1, \ldots, 4$, has been computed by modeling each propeller as a thin disk of mass 0.0039 kg and radius 0.1145 m. Lastly, we set $K_{\mathrm{P},1} = \mathrm{diag}(1, 1, 2, 0.4, 0.5, 0.4)$, $K_{\mathrm{D},1} = \mathrm{diag}(0.5, 0.5,\ 1, 0.1, 0.25, 0.1)$, $K_{\mathrm{P},2} = 0.5 I_6$, and $K_{\mathrm{D},2} = 0.25 I_6$; the adaptive rate matrix Γ and the matrices P and Q that verify the Lyapunov inequality (5.35) have been omitted for brevity. The inertial parameters of the camera have

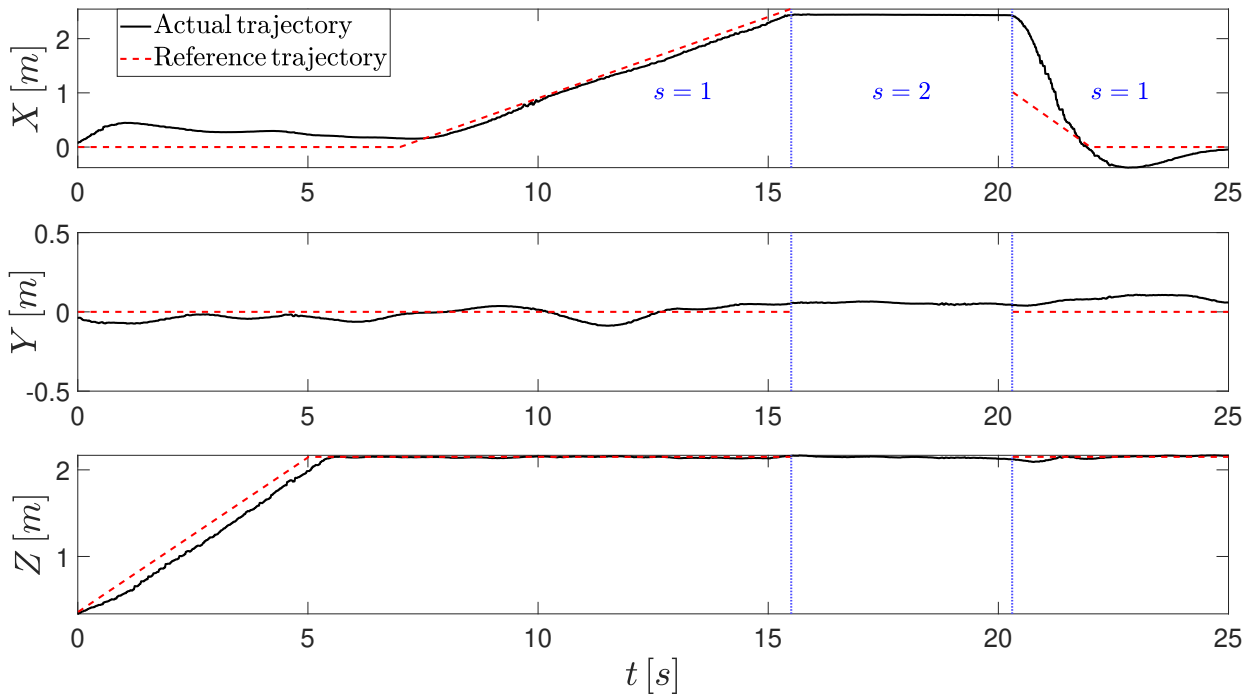

Figure 5.8: Position of the UAV during a sensor placement mission. Both the feedback-linearizing control law (5.55) and the adaptive law (5.30) have been tuned without accounting for the camera. Therefore, during the adaptive gains' transient dynamics over the time interval $[0, 7.2]$ s, the trajectory tracking error is sensibly larger than over the time interval $[7.2, 20.3]$ s. To lead the UAV away from the vertical surface as soon as possible and overcome the force exerted by the suction cup, the reference trajectory is discontinuous at $t = 20.3$ s.

been neglected while tuning the proposed controller.

Figure 5.8 shows the components of both the actual trajectory $r_A^{\mathbb{I}}(\cdot)$ and the reference trajectory $r_{\text{ref}}^{\mathbb{I}}(\cdot)$ of the point $A(\cdot)$. The aircraft takes off at $t = 0$ s and is commanded to reach an altitude of 2.15 m over the time interval $[0, 5]$ s. Successively, the reference trajectory requires the UAV to hover for two seconds. At $t = 7$ s, the reference trajectory leads the reference point $A(\cdot)$ toward the wall at a forward velocity of 0.33 m/s. The wall is impacted at $t = 15.5$ s and the propellers' tilt angles are held constant over the time interval $[15.5, 20.3]$ s to exert sufficient horizontal force to engage the fabric strip fastener. At $t = 20.3$ s, the suction cup's micro-pump is deactivated and the UAV follows its reference trajectory moving away from the vertical surface. The reference trajectory is discontinuous at $t = 20.3$ s to lead the UAV away from the vertical surface as soon as possible and overcome

the residual force exerted by the suction cup. Therefore, in this flight test $t_0 = \tau_0 = 0\,\mathrm{s}$, $\tau_1 = 15.5\,\mathrm{s}$, and $\tau_2 = 20.3\,\mathrm{s}$. Although t_0 and τ_2 have been set *a priori*, τ_1 could be only estimated over the course of the flight test, which provided an additional challenge for the proposed control architecture.

Over the time interval $[0, 15.5)\,\mathrm{s}$, the $\mathcal{L}_\infty$-norm and the $\mathcal{L}_2$-norm of the trajectory tracking error are $0.4820\,\mathrm{m}$ and $0.1904\,\mathrm{m/s}$, respectively. Over the time interval $[15.5, 20.3)\,\mathrm{m}$, the $\mathcal{L}_\infty$-norm and the $\mathcal{L}_2$-norm of the trajectory tracking error are $0.0412\,\mathrm{m}$ and $0.0226\,\mathrm{m/s}$, respectively. Finally, over the time interval $[20.3, 25]\,\mathrm{s}$, the $\mathcal{L}_\infty$-norm and the $\mathcal{L}_2$-norm of the trajectory tracking error are $1.4270\,\mathrm{m}$ and $0.4054\,\mathrm{m/s}$, respectively. The target position $r^{\mathbb{I}}_{\mathrm{wall}} = [2.15, 0, 0]^{\mathrm{T}}\,\mathrm{m}$ is reached with $0.01\,\mathrm{m}$ error at $t = \tau_1$. The proposed control architecture has been tuned without accounting for the camera, which lead to larger initial errors in the trajectory tracking error along both the X axis and the Z axis, which are compensated over the time interval $[0, 7.2]\,\mathrm{s}$. The discontinuity in the reference trajectory $q_{\mathrm{ref}}(\cdot)$ lead to a large trajectory tracking error, which is compensated over the time interval $[20.3, 25]\,\mathrm{s}$.

Figure 5.9 shows the UAV's pitch angle. Over the time interval $[0, 15.5)$ s, the $\mathcal{L}_\infty$-norm and the $\mathcal{L}_2$-norm of the pitch tracking error are 0.0993 and $0.1042\,\mathrm{s}^{-1}$, respectively. Over the time interval $[15.5, 20.3)\,\mathrm{s}$, the $\mathcal{L}_\infty$-norm and the $\mathcal{L}_2$-norm of the pitch tracking error are 0.0430 and $0.0177\,\mathrm{s}^{-1}$, respectively. Finally, over the time interval $[20.3, 25]\,\mathrm{s}$, the $\mathcal{L}_\infty$-norm and the $\mathcal{L}_2$-norm of the pitch tracking error are 0.0423 and $0.0398\,\mathrm{s}^{-1}$, respectively. Since the proposed control architecture has been tuned without accounting for the payload, the UAV experiences a positive pitch angle over the time interval $[0, 15.5]\,\mathrm{s}$. The reaction force produced by disengaging the suction cup from the wall produced a negative moment and hence, a negative pitch angle over the time interval $[20.3, 25]\,\mathrm{s}$.

This chapter presented a model reference adaptive control framework for unknown nonlinear switched plants, whose trajectory of a user-defined linear switched reference model. For the first time, the effectiveness of an MRAC law for unknown nonlinear switched plants

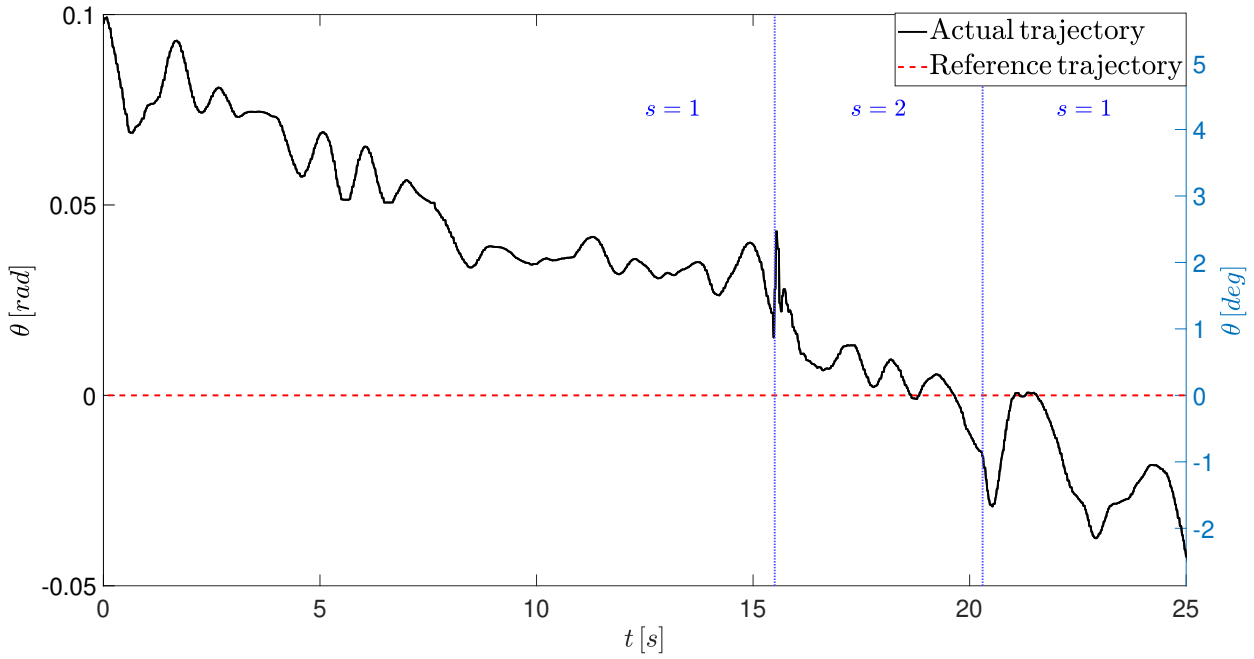

Figure 5.9: Pitch angle of the UAV. The proportional-derivative baseline controller has been tuned without accounting for the camera. Therefore, the UAV experiences a pitch moment due to the gravitational force, which is counteracted by the adaptive control law.

is proven by analyzing both Carathéodory and Filippov solutions. The effectiveness of the proposed results have been verified both numerically and by means of flight tests. The proposed numerical simulation involves the design of an adaptive control law for a reconfigurable delta-wing aircraft, whose aerodynamic and geometric coefficients vary instantly and are unknown. The proposed flight tests involve the design of a control law for a UAV tasked with installing a camera of unknown inertial properties at a user-defined point on a vertical surface.

Chapter 6

An algorithm for using an unmanned aerial vehicle in contact-based inspection of building envelope

The main goal of this dissertation has been to develop a solution to the routing and control problem for multirotor UAVs. After studying the effects of the objective function on the routing problem in Chapter 2, the work in Chapter 3 extended the classical vehicle routing formulation to one more suitable for UAVs by adding battery swap locations and accounting for wind. To ensure the UAVs capability to complete these missions, two MRAC-based control techniques have been developed. The first, developed in Chapter 4, is designed for UAVs that must complete tasks where a priori guarantees on the trajectory tracking error and control input are vital, and the second technique, developed in Chapter 5, ensures stability for a UAV carrying an unknown payload that must interact with a hard surface. The goal of this chapter is combine the ideas presented in the previous chapters into one cohesive application of using UAVs to improve safety on construction sites.

The use of UAVs on construction sites is steadily increasing in recent years. While most of the applications involve the improvement of safety or mapping the site in a passive manner, future UAVs could provide even more data feedback through contact-based measurements to improve safety and real-time data acquisition through the integration with the Building Information Model (BIM). In this chapter, a method for using a UAVs to measure

autonomously the moisture of an exterior precast concrete envelope is developed. To plan the path of the vehicle, an A* algorithm with a custom heuristic, developed in Section 3.1.1, underlies the cost matrix of a modified Traveling Salesman Problem (TSP) which considers visiting all sites, imposes a constraint on the flight time, and considers battery swaps since UAVs have limited flight time. This TSP formulation is a special case of the routing problem developed in Section 3.2, where only 1 vehicle is considered and no delivery constraints are considered. To enable the UAVs to contact surfaces, a switched MRAC law, developed in chapter 5, is employed to control the vehicle and guarantee successful measurements despite discontinuities in the system dynamics. A full physics-based simulation environment is developed, and the proposed framework is used to simulate taking multiple contact-based autonomous measurements, and the data is integrated with the BIM model. A preliminary prototype vehicle has also been developed and discussed.

6.1 Introduction

While the interest in the use of UAVs on construction sites is increasing, there are still some strict rules on how UAVs are to be used in industry. To become a certified UAVs pilot, a candidate pilot must pass the FAA Part 107 test to verify competences in the use of UAVs [195]. Certified must abide by multiple rules, including

1. The UAVs must weigh less than 55 lbs;
2. The UAVs must be visible to the pilot without any visual aid at all times;
3. The UAVs cannot be flown in controlled airspace, in proximity of airports or over people;
4. Only daytime flight is permitted;
5. Flight altitude limit is 400 feet above ground.

Rule (1) can be somewhat limiting as it prevents a construction site from having large vehicles that can lift heavy payloads. Rule (2) is quite restrictive for construction applications. Since the vehicle must be in line of sight of the pilot, one cannot fly around a building without the pilot walking around the building as well. Rule (3) is important when considering the safety of those near the vehicle. Since there are relatively few studies on the safety of UAVs in general, for now the FAA does not allow UAVs to operate overhead of anyone not involved in the flying [195]. This is limiting for construction applications as it would essentially require a mostly empty site to ensure that this rule is followed. The results presented in this paper apply under the following assumption.

Assumption 6.1.1. All experiments are performed with only the pilot and the data collectors located on the site.

Assumption 6.1.1 ensures that we are following the general Part 107 guidelines at all times and to guarantee the safety of everyone onsite. In addition to the basic rules provided by the FAA, there may be rules for any specific construction site. There is a very limited literature that attempts to study the safety of UAVs in construction as safety can be difficult to quantify. The authors in [196] perform risk analysis of UAVs on construction sites where they account for some of the FAA rules such as flying over people not involved in experiments. Further studies should be performed to determine what UAVs operations can be considered as safe and how this should translate into rules for flying vehicles on site.

For the type of experiments proposed in this paper, contact with a surface is required. The ability of the UAVs to perform manipulations and inspections provides a safe assistant to complete the tasks that are dangerous for humans. An example of such tasks is the inspection of the building exterior walls. Water damage is one of the main issues with the building envelope such as in precast concrete façade. The main points of weakness for precast concrete include cracking, fractures, and water damage [197]. While cracks and fractures can be inspected by a camera, taking moisture measurements requires placing the sensor against

Figure 6.1: Left images shows workers at height inspecting and repairing building envelope. The center image shows damages on the precast concrete envelope due to lack of periodic inspection. The right image shows building exterior envelope requiring inspection and repair at height during construction and major renovation.

the wall for a short period of time. As shown in Figure 6.1, the exterior facade can be difficult to reach by human workers, unless a swing-stage or an elevated platform is used but working at height is a hazardous condition and might cause workers to fall causing serious injury or death. According to Occupational Safety and Health Administration (OSHA) falls are among the most common causes of serious work-related injuries and deaths in construction [198]. Being able to utilize a UAVs to perform inspection and repair tasks on the building envelope improves the onsite safety as fewer human workers must subject themselves to hazardous work at height.

In this chapter, an algorithm is proposed for using a UAVs to perform contact-based autonomous moisture measurements on structures containing precast concrete. A Building Information Model (BIM) is converted into a grid that can be used for an optimal path planning technique for determining the best routes between measurement locations. The A* path generating technique, developed in Section 3.1.1, with a customized cost function for safety margins is utilized for determining the best set of possible paths. A heuristic approach is then developed to estimate the best solution to the path optimization problem. Since the vehicle must contact the surface at the measurement locations, the robust switched adaptive

control technique, developed in chapter 5, is implemented to ensure the vehicle is both stable and precise. To rapidly integrate measured data back into the BIM, a script is generated using Dynamo to visualize the data in Revit, which is one of the mostly used BIM authoring tools. A full simulation environment is developed, and the algorithm is numerically tested. Furthermore, a preliminary prototype vehicle is developed for future experiments.

6.2 UAVs to improve construction safety

Presently, the most common use of UAVs in the construction industry falls under the umbrella of visual data acquisition including safety inspection and monitoring. Specifically, there have been many examples of UAVs being used to perform safety inspections on sites, see [199, 200, 201, 202]. UAVs have been used to monitor and perform visual inspections of difficult to reach locations such as skyscrapers or cranes [203]. One of the first to include UAVs that can touch the environment is presented in [204] where a UAVs holding a sensor touches the wall. In addition to just taking pictures, algorithms have been developed to analyze the site for deformations and perform infrastructure evaluation to predict future failures [205]. In [206], it is noted that UAVs are being used to perform safety checks and monitoring the sites for safety violations. This type of monitoring for safety violations is generally categorized into two types; real time or trigger-based. In real time monitoring, the area is always under surveillance, whereas in trigger-based monitoring, the vehicle is given a signal of when to take off and begin surveying the area, such as after an incident has occurred [207]. Image processing-based safety hazards identification techniques, like the one proposed in [208], can be used with UAVs for site monitoring and early detection of any safety hazards. A survey conducted with safety managers found that the potentially most important safety improvement jobs for UAVs include monitoring boom vehicles or cranes in the proximity of overhead power lines and monitoring unprotected edges or openings [209].

In addition to the improvement of safety by using UAVs, the safety cost of using UAVs should be considered. Currently, UAVs users in the construction industry are to follow a set of safety regulations by OSHA which are based on the Part 107 regulations set by the FAA [210]. In an attempt to quantify the risk to workers, researchers in [196] performed quantitative and qualitative analysis using simulations and estimated five potential fatalities every one million flight-hours. While not negligible, it is considerably less risky as compared to other risks associated with construction works at height.

An interesting issue in regards to UAVs safety is that different UAVs can have different margins of safety. In fact, the authors in [211] found a correlation between the cost of a UAVs and its embedded level of safety. Furthermore, a recent survey of safety managers found that the most important characteristics of a UAVs for improving onsite safety are the ability of an onboard camera to be manoeuvrable by the user, detect and avoid capabilities of the vehicle in the environment, and a real-time video communication feed for live updates [209].

6.3 UAVs for improving BIM

BIM is a data rich multidimensional model-based process that gives construction professionals, architects, and engineers the ability to plan, design, construct, and manage buildings more efficiently. The classical BIM involves 3-dimensional modeling where the building or structure can be fully modeled including listing all dimensions of the building, the building geometry, all building components involved, and much more. One of the main benefits of such a model is the speed and accuracy in which information can be shared within the parties involved in the building's lifespan such as architects, engineers, owners, and constructors [212, 213].

One of the most challenging stages of the build process in regards to BIM is the con-

struction phase. Researchers have shown the benefits of utilizing BIM include, but are not limited to, increasing information available to support upstream design decisions, reduction of man hours, and increased communication throughout the project. BIM achieves these results by attempting to synchronize the design and construction planning and linking the construction process to a simulated model to predict and show what the building may look like at any stage in the process. In addition to the construction phase, BIM is also useful in the "as built" phase after project completion. In this scenario, BIM model can alert owners or engineers of issues or predicted issues with the structure, and the virtual model can assist in rapidly assessing the situation and developing a solution strategy. Digital twins are used for performing sustainability assessment of existing buildings [214]. Figuring out how to update BIM models in real-time, or close to real-time, is one of the processes' most challenging tasks.

Recently, UAVs have been utilized to integrate information about the site into the BIM models. One of the more common practices for using UAVs in conjunction with BIM is for progress monitoring. In many cases, the UAVs is tasked with taking pictures or laser scans of the site [69, 70, 71]. In these cases,the images and scans are post-processed in order to update the BIM to include time parameterization with real site progress. UAVs performing the scans and taking measurements is not only safer due to not having to deal with working at height, but it is shown to be both faster and cheaper in some cases even after including the post-processing time[215]. Going one step further, the authors in [216] utilize UAVs to search the buildings and identify missing or new electrical outlets, and then the BIM would automatically be updated with this new information. A UAVs was used to update BIM in real-time on a water pipeline project where the virtual environment and UAVs were integrated such that a visualization of the UAVs's view is projected into the simulated environment. This also allows direct comparison of the UAVs view of the real environment with the virtual BIM [216]. It can be noted, that according to our literature review, none of the data gathered by UAVs on construction sites has required direct physical

contact with the building itself.

6.4 UAVs for contact-based tasks

Almost all of the previously listed applications of UAVs have accomplished their respective missions without any part of the vehicle having to physically touch hard surfaces within the environment. However, in the last decade, there has been a significant increase in the area of using conventional UAVs for more than just camera-based measurements. In [217], a group of UAVs autonomously assembles a rope bridge within a motion capture space. The authors in [73] used a thrust-vectoring UAVs to pull a cart attached with a rope. Another application of thrust-vectoring UAVs has been their use to push or maneuver boxes around a room [218, 219].

In addition to more conventional UAVs, there have been additional research efforts in outfitting UAVs with an actuated robotic arm. These vehicles are commonly known as Unmanned Aerial Manipulators (UAM), and these aircrafts can also be used for interactive tasks. UAVs, when equipped with a robotic arm, have capabilities to push, pull, grab, drop, grab and turn, push and slide, functioning as a UAM. These capabilities have been used for assembly tasks, installation and retrieval missions, tool operations, and valve turning etc. In fact, UAMs have been used for installing and retrieving sensors [220], structural inspection by contact [204, 221], constructing simple truss like structures [222, 223], building brick towers [217], and some more complex tasks like peg-in-hole and valve turning tasks [224] etc. These UAVs are equipped with additional components in various ways to enable them to interact with the environment. For example, [220] used a prismatically joined active manipulator, that has the capabilities to push and pull, and have been tested for installing sensors in both indoor and outdoor environments. The authors in [221] used the Sensima UPec eddy current inspection kits with UAMs to push and slide over metallic surfaces, to detect welding

joints. Although they tested their system in a controlled environment, their research has presented an opportunity to use this system for structural inspections of containers or any metallic surface in gas and oil industry. The researchers in [225] developed a system that deposit liquid expansion foam to repair cracks in metallic surfaces. Furthermore, researchers in [222] and [217] used the grab and drop capability of the UAMs to construct simple truss like structures and brick towers.

These examples represent potentials for the UAMs to be used in situations that require human workers to be transported to the target locations on construction sites using cranes or being suspend from the building structure through harnesses, both of which expose workers to higher risk of injury. These procedures are costly and time-consuming, but most importantly put human workers in hazardous situation making them vulnerable to accidents. The ability of UAMs to perform simple manipulations and inspections provides an alternative to complete tasks fast and less hazardous to human workers safety.

6.4.1 Control of UAVs for contact-based tasks

UAVs that touch or interact with items or surfaces in the environment provide additional challenges to the already difficult nonlinear control problem. Many, if not most, construction applications require the UAVs to operate outdoors. In this case, wind becomes a factor, and it is known that wind can cause some strong disturbances near buildings or vertical surfaces which can be difficult to model exactly [226]. This provides a need for a control law that is robust to unknowns. Some examples of nonlinear robust control to reject disturbances for UAVs include a robust backstepping control based on integral sliding mode control to handle wind gusts [227], hierarchical control to compensate for input delays and parametric uncertainties [228], robust model reference adaptive control to account for bounded external disturbances such as wind [229], a robust sliding mode control law to account for nonlinear disturbances [230], among others.

One of the potential applications of UAVs in construction and major renovation involves transporting goods within the site. The payload for each pickup and delivery may be different, even unknown, and an unknown or unsteady payload contributes to uncertainty in the system. Additionally, a payload may shift the center of mass of the system, and since it is customary to use the UAVs center of mass as the vehicle's reference point, then the UAVs translational and rotation equations of motion become tightly coupled. [231]. Some examples of robust control laws for UAVs to handle unknown payloads include an adaptive control method [232], geometric control [233], heirarchical control [234], energy-based control [235], linear quadratic gaussian control [236], linear quadratic regulator control [237], to name a few. Accounting explicitly for a changing center of mass and matrix of inertia, the authors in [73] developed a robust model reference adaptive control law to allow a UAVs to pull a cart or transport heavy sling payloads.

For UAVs scenarios involving sensor measurements that require contact with the surface, there are significant modeling and control challenges. The impulsive forces originating from wall contacts can cause the dynamics of the system to be discontinuous. To address the problem of controlling discontinuous aerial systems, some authors have resorted to using the hybrid control framework or switched systems control framework. The authors in [238] used the hybrid control framework to allow for a UAVs to perform contact measurements. The hybrid framework is also used in [239] wherein a quadrotor must maintain stability when required to dock and undock with a vertical surface. In chapter 5, an MRAC law for switched dynamical systems has been developed to allow for a thrust-vectoring quadrotor to autonomously mount a camera onto a wall.

6.4.2 Path planning for UAVs in construction

There are several challenges in path planning and navigating on a construction site for UAVs. Typically, unstructured and dynamic environments will require active obstacle avoid-

ance and continuous re-planning. Replanning itself may be difficult if the highly nonlinear dynamics of the vehicle are taken into account, and very few optimization-based techniques can run in real-time. Additionally, although the path planner generates obstacle-free paths, the quality of localization, or accuracy of the location of the vehicle with respect to the environment can be poor. Most current applications require a Global Positioning System (GPS) to capture the current location of the aircraft. However, GPS is typically only accurate to within about 0.5m. While this is accurate enough for some basic image and video gathering missions, higher accuracy is required for precise measurement gathering, specifically in construction tasks including contact-based inspection of the building envelope. Typically GPS commonly contains positioning error of up to 1 meter, whereas differential GPS can have much better accuracy but is also significantly more expensive. The recent rise of Real-Time Kinematic (RTK) GPS, which can have an accuracy of less than 2cm [240], could help alleviate the localization problem. This method requires to have a base station to be installed on location, and a follower receiver is then mounted on the vehicle. Both GPS and RTK can lose accuracy in proximity of reflective surfaces, and they require a clear view of the sky for best use [241].

While there are some answers for the localization issue on sites, there still remains a larger issue of the dynamic environment. Very rarely is a building site static for very long, if at all. There are moving people, trucks, cranes, and the building will change shape and size over the course of its construction and major renovation. For this reason, path planning may fail if it leverages on pre-recorded data and a real-time collision avoidance algorithm may be necessary. Recently, researcher have successfully navigated through cluttered and unknown areas [242, 243], but these results are all still applicable within slow moving environments.

Assumption 6.4.1. We will assume that the environment onsite is static for the entire mission and that either an accurate CAD model or point cloud of the site is available for planning purposes.

The main necessity of Assumption 6.4.1 is the need for a known map for global path planning to guarantee collision-free paths. In many construction cases, a well-developed BIM is available, and it is shown in Section 6.5.1 how we convert the BIM to a usable map for planning. While the static map assumption can be relaxed if the map can be updated in real-time, including a live mapping algorithm with full camera integration is beyond the scope of this work. Previous researchers have implemented cameras and or LIDAR scanners on small UAVs for accurately mapping sites and could potentially be used in a companion UAVs along with the proposed technique by developing a proper real-time communication protocol [244, 245, 246, 247]. Furthermore, the inclusion of a dynamic map introduces the need for an additional collision avoidance algorithm. This type of algorithm requires multiple aspects, namely a real time map update to deduce obstacles and a real-time trajectory planner with embedded collision avoidance. Integration of these techniques for highly dynamic environments is an active field of research in robotics and is to be explored and integrated in future work.

Determining the best flight path for a UAVs in a construction application is heavily dependent on the desired application. For applications involving scanning an entire building or capturing images of the structure, it is most important to make sure that all the areas around the site are visited. An approach to solve this problem is known as the strip method [248, 249]. This method typically involves stitching together horizontal flight paths together and works best assuming a low flight speed [250]. The main challenge for this method is to determine the desired overlap of images from each horizontal pass of the structure, and this determines how far apart the horizontal paths are placed vertically from each other [71]. Another approach to flight path design involves determining the next best view for each waypoint in the flight path [251]. Other authors take time of day into account to ensure that lighting does not affect the quality of the desired photos or measurements [246, 252]. Another popular method of path planning for construction applications involves using a GPS and setting desired waypoints for the UAVs. The authors in [253] used a system of waypoints

which were preloaded onto the vehicle before the mission. In [246], the researchers connected a computer to a ground station software that allowed to select the desired waypoints which were stored in an XML file. The UAVs then visited these waypoints while being localized by a differential GPS.

6.5 Developing a UAV to measure moisture content in precast concrete

To test the idea of taking contact-based measurements in construction and major renovation that could be used to update BIM models, we developed a UAVs that can use a pinless moisture meter to take moisture measurements of precast concrete exterior walls. Precast concrete is advantageous to the ability of rapid build times for precast concrete structures, and entire buildings can be formed from it.While precast concrete is resistant to water and air penetration, there is a weakness in the panel joints when multiple individual joints are put together [254, Ch 2.]. Water damage is one of the main issues with precast concrete, and if detected by measurements in the early stages, it can be corrected [255]. Moisture measurements can be taken on precast concrete by holding a sensors against the wall. In cases of practical interest, moisture measurements need to be taken at locations that are difficult to reach, unless a swing-stage or an elevated platform is used. Being able to utilize a UAVs to take these types of measurements improves the overall safety onsite as fewer persons must subject themselves to risky measurements at high locations.

Figure 6.2 shows the key components to our proposed approach for autonomous measurements. The user inputs include a map of the environment and the locations to take the measurements. A conversion algorithm is proposed to convert the BIM and measurement locations to an occupancy grid that can be used for path planning. The path planner is made up of two parts. The first is the path generator, developed in Section 3.1.1, which leverages

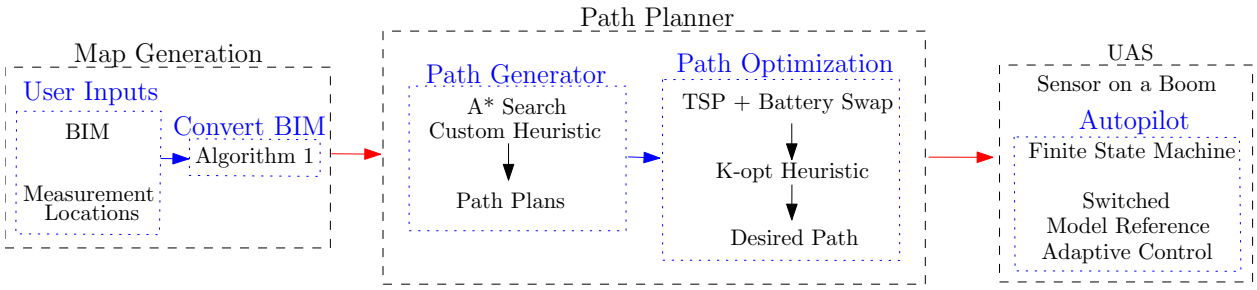

Figure 6.2: Overall structure of the proposed autopilot. The user inputs a map or building model and desired measurement locations. The path planner is made up of two parts. A path generating algorithm using A* search finds shortest paths between all of the locations while maintaining safe distances from the building. These paths are fed to a path optimization algorithm based on a Traveling Salesman Problem which then determines the shortest path which visits all locations. The UAVs carries the sensor on a boom. The autopilot features a finite state machine to switch autonomously between flight modes, and an adaptive control law ensures mission completion despite modeling uncertainties and external disturbances.

the known map and uses an A* search algorithm with a custom heuristic to account for energy consumption and maintain safe distance from the building. The path optimization algorithm approximately solves a Traveling Salesman Problem (TSP) with a heuristic to determine the fastest mission time. The TSP a vehicle is required to visit a pre-defined set of locations while obeying constraints. Another example of a combined A* and TSP path planner used in building inspection can be seen in [256], where the authors develop a novel approach to performing non-contact inspection of trusses. Constraints in this case include that all desired locations much be visited, the UAVs has a limited range and may have to revisit the starting location to swap batteries, and each location will have a time required to take the measurement. This algorithm also accounts for the potential case where the vehicle must return to the start and swap batteries during its mission. The desired path for the mission can then be uploaded to the UAVs which is a quadrotor with a sensor mounted on a boom. The onboard autopilot is based around a finite state machine which uses logic to switch autonomously between different flight states such as takeoff, waypoint navigation, sensor measurement, or return and land. The switched MRAC law from chapter 5 is used to ensure the UAVs can complete its mission despite disturbances such as wind and the reaction forces and moments introduced by the sensor contacting the wall.

Figure 6.3: Image of Bishop-Favrao Hall which is utilized as a test case for this work.

6.5.1 Map generation

For the path planner in this work, a map in the form of an occupancy grid is needed. An occupancy grid in this case is a binary, uniform discretization of the environment such that all occupied cells are given a value of 1, and free cells are given a value of 0. This section describes how the BIM model can be converted into an occupancy grid.

To test the UAVs, the BIM model for Bishop-Favrao Hall (BFH) located on the Blacksburg campus of Virginia Tech is used. Autodesk Revit is the most-commonly used BIM authoring tool in the Architecture, Engineering and Construction (AEC) industry and same has been used in this research for modelling the BFH. BIM models are object-oriented, but to create occupancy grid for automated maneuvering of the UAVs, geometric information from BIM has to be converted into a CAD format. For this the BIM model from Revit is converted into the StereoLithographic (STL) format, sometimes also known as Standard Tesselation Language. Autodesk's STL Exporter for Revit 2020 add-in was used to convert the BIM model into an STL file.

For the path planner in this work, an occupancy grid is needed, and this section describes how the STL file from Revit can be converted into an occupancy grid. Consider the points of the STL file as $p_i \in \mathbb{R}^3$, $i = 1, \cdots, n_p$, where n_p denotes the total number of points. Let

$C \in \mathbb{R}^{n_c \times 3}$, where n_c is the number of connections, and each row in C corresponds to a triple of integers denoting the indices of points in the list which create the edge connections between different points. All of the space within the boundaries of these connected points is occupied.

We can first add all of the STL points to the occupancy grid as occupied. However, we need to account for the interior points that would be located within the regions generated using the list of connections in C. Each row in the connectivity list, C, provides three indices, i, j, k, which are integer values between 1 and n_p, and these indices correspond to three points, p_i, p_j, p_k. These points, which form a triangle with area λ_{ijk}, are first rounded using the grid resolution. Then a list of potential interior candidates are generated. These test points are created by making a cube from the smallest to the largest x, y, z coordinates within the current set of vertices.

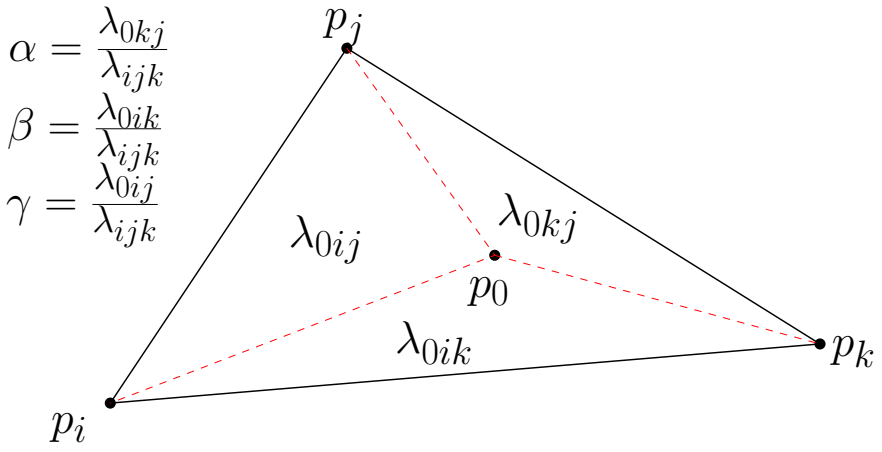

Figure 6.4: Diagram for determining if a point lies in the interior of a triangle. By using the vertices, p_i, p_j, p_k and the sample point, p_0, it can be determined if the sample point lies inside the triangle. The λ_{ijk} values represent the area of the triangle generated using points p_i, p_j, p_k.

To check if a test point, denoted by p_0, lies in the plane generated by the connections, we first project the current test point onto the plane containing the points p_i, p_j, p_k.Then using barycentric coordinates and following the methods in [257, Ch. 2], it can be verified if that point is in the convex hull generated by this triangle [258, Ch. 2]. Let δ_{ij} denotes the line

segment between vertices p_i and p_j, then the area, $\lambda_{ijk} > 0$, of the triangle is given by

$$\lambda_{ijk} = \frac{|\delta_{ij}^{\times} \delta_{ik}|}{2}, \tag{6.1}$$

where $\| \cdot \|$ denotes the vector norm and $(\cdot)^{\times}$ denotes the cross product operator. Then, the area of 3 triangles are considered by using the test point, p_0, along with combinations of points p_i, p_j, p_k. The first triangle is comprised of vertices located at p_0, p_j, p_k and has an area, denoted α, which is normalized by λ_{ijk}. The first triangle is comprised of vertices located at p_0, p_i, p_k and has an area, denoted β, which is normalized by λ_{ijk}. The third triangle, comprised of vertices p_0, p_i, p_j, has normalized area γ. These areas are found using the following equations,

$$\alpha = \frac{\|\delta_{0j}^{\times} \delta_{0k}\|}{2\lambda_{ijk}}, \tag{6.2}$$

$$\beta = \frac{\|\delta_{0k}^{\times} \delta_{0i}\|}{2\lambda_{ijk}}, \tag{6.3}$$

$$\gamma = 1 - \alpha - \beta. \tag{6.4}$$

If $\alpha \in [0, 1]$, $\beta \in [0, 1]$, $\gamma \in [0, 1]$, and $\alpha + \beta + \gamma = 1$, then the point p_0 lies in the triangle generated by p_i, p_j, p_k, and p_0 is considered as an occupied point and added to the obstacle list. This process is repeated for every set of connections in C. Figure 6.4 shows a graphical representation of the method. A pseudo-algorithm of the STL map conversion is given in Algorithm 6, where the output of the algorithm is the map to be used in the path planner.

6.5.2 Path generator

The first goal of this work is to have our UAVs visit a set of user-defined locations, or waypoints, within the site as quickly and efficiently as possible while maintaining safe distance from the building between locations. These locations define the points on the

surface at which the measurements will be taken. For our path planner, we are not trying to complete a full scan of the building structure and many existing techniques do not apply. We are assuming that the location is well-mapped, and there will be no additional obstacles in the environment other than those already in the available model. In many cases, a BIM is available to provide the accurate model, but other techniques such as using a UAVs with cameras or LIDAR can be used to generate accurate models [244, 245]. To achieve the path planning requirements, a two step process is presented. In the first step, a set of paths are generated using an A* search between all possible pairs of waypoints while maintaining safe distance from the building. Secondly, we use an optimization-based approach to select the optimal subset of the A* paths which can be seen as an extension of the classical TSP, and a special cause of the optimization approach in Section 3.2. To calculate the travel times between the different locations in the problem, the following assumption is used.

Assumption 6.5.1. The vehicle is assumed to fly at a constant speed between all waypoints.

Assumption 6.5.1 is used to calculate the total mission time to guarantee the feasibility of the proposed paths. Another approach could be to more conservatively assume a lower bound on the velocity as a precaution to ensure that the total time in the air is less than the battery endurance. An approach to relax this assumption in the future is to implement a trajectory planning technique that calculates the full state vector at each instant and the total time between waypoints would be available directly. However, this is a computationally demanding approach that can be difficult to implement in real time in larger environments.

To generate all of the possible paths between the nodes, we propose to use an A* algorithm to find the shortest paths between each pair of nodes. A basic description and sample algorithm can be found in [65], and fundamentally the idea is to find the set of neighboring nodes that connect the start node to the goal node with the lowest cost. To ensure additional safety from hitting the building, it is desirable to push the paths away from the building to a user-defined safety margin. A custom heuristic weighting function, inspired by [64], is

implemented to accomplish this goal. This function makes nodes that close to the building have a very high cost, and the function decays to the classical Euclidean distance heuristic as the distance from the wall approaches to a user-defined value.

The full A* algorithm is described in Section 3.1.1 and briefly reviewed in the following section for convenience. Consider the map, $\mathcal{M}$, with the set of nodes $n \in N$ and arcs $a \in A$. The initial node is denoted n_0, and the goal node is denoted n_{goal}. For a given node n, all nodes adjacent, or connected to n, are considered as neighbors. In our case, the graph is fully connected, and any set of nodes can be considered as adjacent, so it is a tunable parameter for us to determine how many nodes are considered as neighbors to any 1 node. Consider the neighbors of a given node n as the set of nodes which are 1 node distance away. Let $\mathcal{O}$ denote the set of obstacles, or the points in the map that are considered as occupied. We define the distance function as $\mathrm{d}_2(n_k, \mathcal{E})$ as the distance between the current node n_k and the set $\mathcal{E}$. In this path generator, we let $\mathcal{E} = \mathcal{O}$ to measure the distance to the nearest obstacle or building.

The *cost function* for the proposed A*-based path planning is given by

$$f_k \triangleq g_k + h_k, \qquad k \in \mathbb{N}, \tag{6.5}$$

where

$$g_k \triangleq \sum_{q=1}^{k} \left[\kappa(\mathrm{d}_2(n_q, \mathcal{O}))\mathrm{d}_2(n_q, n_{q-1})\right], \tag{6.6}$$

denotes the *cost-to-come function*,

$$h_k \triangleq \mathrm{d}_2(n_k, n_{\text{goal}}), \tag{6.7}$$

denotes the *heuristic function*,

$$\kappa(\alpha) \triangleq 1 + \frac{\mu_1}{\mu_2 + e^{\mu_3 \alpha}}, \qquad \alpha > 0, \tag{6.8}$$

denotes the *weighing function* where $\alpha > 0$ is the distance from the current point to the nearest obstacle, and $\mu_1, \mu_2, \mu_3 > 0$ are user-defined parameters. For additional details on A* path planning in the presence of obstacles, see [64]. This function has a maximum at $\alpha = 0$, and hence strongly discourages the path planner from producing points close to the obstacles. The function is strictly decreasing and tends to 1 as $\alpha \to \infty$ which suggests that obstacles that are far away from goal points have little influence, and we get a shortest path behavior arbitrarily far from the building.

To demonstrate the validity of the proposed approach for this application, a CAD model of BFH has been imported using Algorithm 6 with a resolution of 0.5m, and A* has been used to plan paths between 7 different user selected locations around the building and a takeoff point located at $(-20, 10, 2)$m, see Figure 6.5. For the weighting function, the parameters have been set to $\mu_1 = 50$, $\mu_2 = 10$, $\mu_3 = 2$, which has a maximum of 5.50 at $\alpha = 0$, and at $\alpha = 4$, the function has a value of 1.01, so the proximity to the wall is minimally penalized for nodes further than 2m from the wall. As seen by the figure, vertical altitude changes are penalized significantly, and the paths only go over the building if it is significantly shorter than going around.

6.5.3 Path optimization

In this section, an optimization approach is developed to determine the best subset of the paths generated in Section 6.5.2 that allow the UAVs to visit all locations as quickly as possible. The problem of determining the lowest cost when requiring a single vehicle to visit a set of locations can be seen as a TSP [52]. In this variant, not only must all measurement locations, be visited, the vehicle must obey a range constraint to ensure the battery is not completely depleted while the vehicle is still airborne. In addition, the option to return to

any battery swap location is allowed. This can be seen as a special case of the optimization problem presented in Section 3.2 by letting the delivery demands be equal to zero, the service time is assumed a known constant at each location, the number of vehicles is equal to 1, and the mass of the vehicle is constant with zero payload.

The set of all nodes is denoted as N' which contains the home location or origin, denoted 0, the set of measurement locations is denoted as M, the set of battery swap locations is denoted as N_b. One approach, commonly used in the Green Vehicle Routing Problem [50], is to consider n copies of the battery swap locations to keep the binary condition on the state variable. For this reason, let N_b' be the set of all battery swap locations and their copies. This problem consists of finding the solution for the binary decision variable, r_{ij}, where $(i,j) \in A$ belong to the set of arcs connecting all measurement and battery swap locations, and $r_{ij} = 1$ if the vehicle travels from location i to location j, and $r_{ij} = 0$ otherwise. Likewise, the cost to go from node i to node j is $c_{ij} > 0$ where we use the time in seconds as the cost. Furthermore, $\tau_j \geq 0$ for $j \in N'$ denotes the time at which the vehicle arrives at node j. $T_{\max}$ denotes the desired maximum time for mission completion and is also used to eliminate subtours, known as the Miller-Tucker-Zemlin (MTZ) formulation, in the optimization problem [72]. The variable b_j, $j \in N'$ tracks the battery capacity at node j, and $e > 0$ denotes the endurance of the vehicle in seconds. The variable m_J for $j \in M$ denotes how many seconds it takes for the vehicle to take a measurement at node j.

Consider the optimization model,

$$\min \sum_{(i,j)\in A} c_{ij} r_{ij} \tag{6.9}$$

$$\sum_{j\in N'} r_{ij} = 1, \quad i \in M, \tag{6.10}$$

$$\sum_{j\in N'} r_{ij} \leq 1, \quad i \in N_b', \tag{6.11}$$

$$\sum_{i\in N'} r_{ij} = \sum_{i\in N'} r_{ji}, \quad \forall j \in N', \tag{6.12}$$

$$0 \leq \tau_0 \leq T_{\max}, \tag{6.13}$$

$$\tau_j \geq \tau_i + (c_{ij} - m_j) r_{ij} - T_{\max} r_{ij}, \quad \forall i \in N', \forall j \in N' \setminus 0, i \neq j, \tag{6.14}$$

$$c_{0j} \leq \tau_j^k \leq T_{\max} - (c_{j0} + m_j), \quad \forall j \in N' \setminus \{0\}, \tag{6.15}$$

$$b_j \leq b_i - c_{ij} r_{ij} + e(1 - r_{ij}), \quad \forall i \in N' \setminus \{0\}, \forall j \in M, i \neq j, \tag{6.16}$$

$$b_j = e, \quad \forall j \in R' + \{0\}, , \tag{6.17}$$

$$b_j \geq \min(c_{j0}, (c_{jl} + c_{l0})), \quad \forall l \in R, \forall j \in M, \tag{6.18}$$

where Eqn. (6.9) denotes the cost function, (6.10) ensures that all measurement locations are visited, constraint (6.11) ensures that each battery swap location is visited at most once, constraint (6.12) captures the standard flow conservation constraints, constraint (6.13) ensures the mission is completed before the maximum allowed time, constraint (6.14) determines the time at which each location is visited, constraint (6.15) ensures the timing at each location is feasible, constraint (6.16) tracks the current battery capacity at each location, constraint (6.17) ensures the battery capacity is set to full at each battery swap location, and constraint (6.18) ensures that the next location will not result in a battery capacity that is too low to reach either a battery swap location or the home location.

The distances between waypoints are calculated by traversing the A* paths. The cost matrix is then calculated by converting the distances between the waypoints to time, using the constant velocity assumption, Assumption 6.5.1. The cost matrix for this problem is based on a constant velocity assumption, Assumption 6.5.1. We consider the UAVs as least efficient in hover or holding position for low speed routing problems [52], and we will assume the battery drain in this worst case scenario that the vehicle is always in hover. A derivation of the power required for a UAV to hover is given in Section 3.1.3. The energy required to traverse the paths is given by the power times time, and since the power is captured by the

hover power and the cost matrix includes the travel times, the relationship between the cost matrix and the remaining flight time is established.

In general, this problem is NP-hard, and can be difficult to solve in real-time, so we use a local search algorithm which utilizes a k-opt procedure [259] and adds or removes refueling trips to minimize mission time. First, we construct an initial feasible solution using Algorithm 7. This procedure starts by initializing the route to the origin. Then, the nearest neighbor method [260] is used to add the next measurement location to the route. The endurance is checked for the current route, and if needed, the cheapest refueling option is added to the route. This procedure is repeated until all measurement locations are visited.

After an initial solution is constructed, the route is improved through a local search heuristic. In this heuristic, the solution is randomly modified by one of two procedures, namely a k-opt procedure and a remove refuel procedure. The k-opt procedure randomly swaps the position of k of the nodes in the current vehicle route. The feasibility of this temporary route is checked, in the same manner as the initial constructive procedure, and if needed a refuel procedure is added. The remove refuel procedure randomly removes one of the refuel stops in the route, if any are available. This temporary solution is then checked for feasibility. The temporary solution from one of the two operations is then used to calculate the cost of this solution, and if the new solution is better than the previous best, this solution is now considered as best. This procedure is repeated until the solution has not improved for a user-defined number of iterations. Algorithm 8 gives a pseudo-code of the local search method.

To test the heuristic procedure, the same locations are used as in Section 6.5.2, see Figure 6.5. The variable $num\_iterations$ is set to 100, $m_j = 30s$ for all measurement locations to include time for orienting perpendicular to the wall, approaching the wall, and holding on the wall for 10s. The maximum endurance is set as $e = 300$s. While most UAVs have endurance longer than 300s, the value is used in this case for emphasis on the algorithms capability of

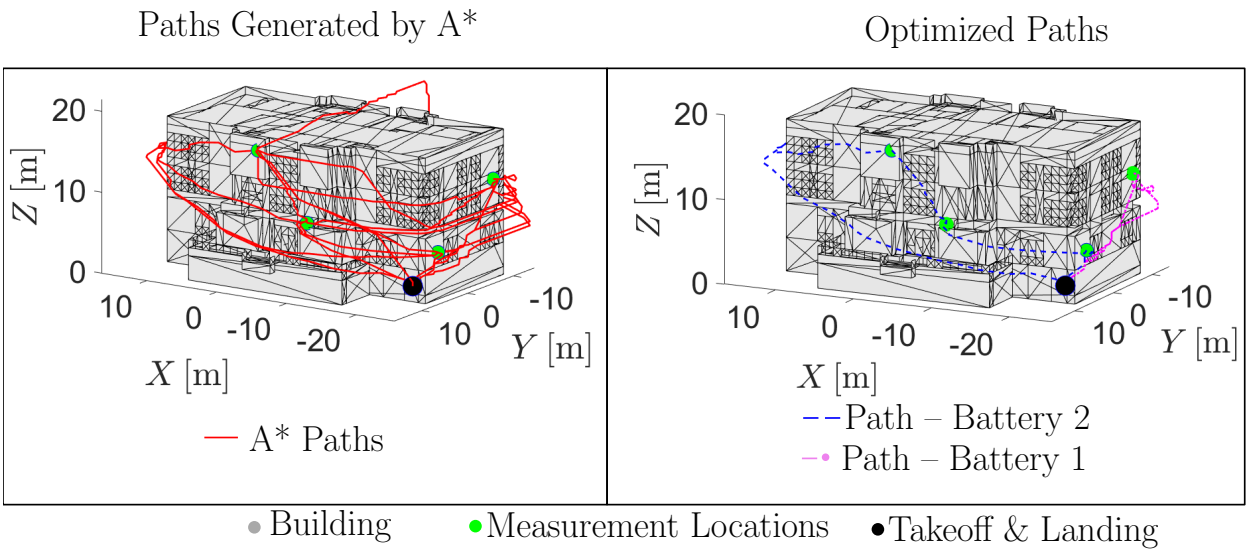

Figure 6.5: Outputs of the path generator and path optimization algorithms. The left figure shows all of the paths between the 7 measurement locations, in green, and the takeoff location, in black. In the figure on the right, the output of the optimization is shown. To complete the mission, a battery swap is needed, and the blue path takes 3 measurements, and returns to the takeoff point for battery swap. The dash-dotted path, in magenta, reaches the other 4 measurement locations before returning.

planning for battery swaps which are necessary for larger instance cases. The right plot in Figure 6.5 shows the output of solving the A* problem and using the generated cost matrix with the $k-opt$ heuristic to find the corresponding shortest path. As can be seen in the figure, two battery packs are needed to complete the scenario. First, 3 measurement locations are serviced, including the one that is farthest from the takeoff point. Next, the other 4 measurement locations are visited, and the path returns to the takeoff point. The algorithm presented in this paper heavily penalizes rising in altitude for two main reasons. The first is increasing in height requires more thrust and is less efficient than moving horizontally in most cases. Secondly, not having the UAVs fly over the building unnecessarily can reduce potential incidents and sudden wind gusts above the building. The total mission takes 507s including a 30s time for changing batteries.

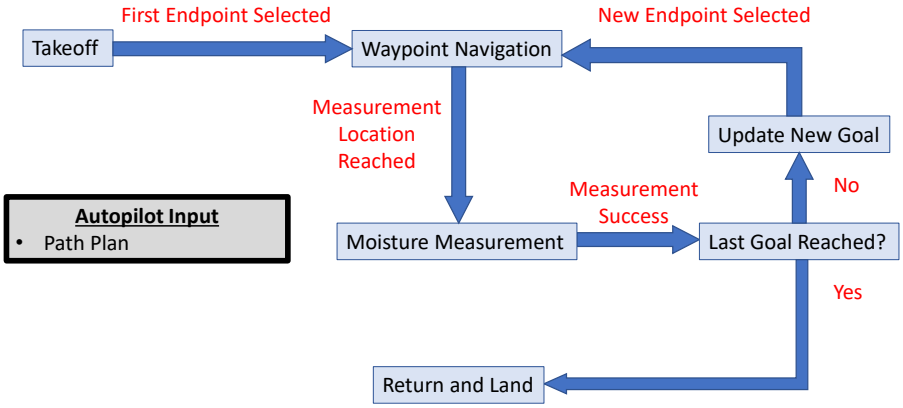

Figure 6.6: Logic architecture governing the current flight state of the vehicle.

6.5.4 Finite state machine

The desired flight path calculated in Section 6.5.3 provides the best path to reach all of the measurement locations, but this path does not consider the takeoff phase, the measurement acquisition phase, or the return to land phase. To enable the UAVs to execute the desired flight path and take all of the measurements, a finite state machine approach is presented to govern the flight states of the vehicle. Figure 6.6 shows the logic architecture of the proposed autopilot. The data import and path planner in Sections 6.5.1 and 6.5.2 are executed offline, and the flight path is uploaded to the vehicle.

A standard flight proceeds in the following manner. First, the vehicle enters the takeoff phase, and the UAVs ascends to a user-defined altitude. Then, the first waypoint is taken from the output of the optimization algorithm, and the UAVs follows the set of points from the corresponding path plan. At any time, if the UAVs is less than a user-defined $\epsilon > 0$ distance from the next point, then we consider that point as reached, and we move to the next point. Once the UAVs has reached the measurement location, or the last point within a single path, the vehicle enters the measurement state. To take a measurement, the vehicle aligns the boom holding the sensor perpendicular to the surface. We assume that both the vehicle's heading and the normal to the surface are known at all times. Once aligned, the

reference path will move towards the wall with the final desired point located just beyond the surface, approximately 0.5m, to ensure the vehicle is exerting force to hold the sensor against the wall. After a user-defined measurement period, same as m_j in Section 6.5.3, the vehicle will then move backwards to the waypoint. If this measurement location is the last, the vehicle returns to the takeoff location and lands. If this measurement location is not the last, then the next location is selected, and the vehicle returns to the waypoint navigation state.

6.5.5 Switched model reference adaptive control

To control the UAVs both along the flight path and during the wall contact phase, we choose to use the switched MRAC law developed in chapter 5, published in [261], which is designed for switched dynamical systems. In our case, the vehicle has two main flight states. The first is when the vehicle is in free flight, and the second is when the vehicle is engaged with the wall. Due to the instantaneous change at the wall contact point, this can be considered as a switched dynamical system in a similar manner to the tiltrotor UAV modeled in Section 5.5.2.

Assumption 6.5.2. The locations to measure the moisture are located on vertical, or very close to vertical, surfaces.

Assumption 6.5.2 is utilized for the simplicity in the vehicle and control design. The vehicle in this paper is a standard quadrotor format which underactuated, wherein the vehicle has to change orientation to move in the horizontal plane. The sensor on this vehicle is mounted at a predetermined angle assuming the wall will be vertical. Mathematically, this assumption allows for the dynamics during wall contact to be a partial state equilibrium of the free flight dynamics and allows for the switched MRAC approach. This assumption can be relaxed in the future by actuating either the sensor mount or the propellers on the vehicle, extending it to a tilt-rotor format.

Almost all off the shelf autopilots contain a Proportional-Integral-Derivative (PID) control law. PID is very computationally efficient and easy to implement, however, it can have significant drawbacks. PID is a linear control technique suitable for linear dynamics, and it has a very limited robustness to uncertainties, that is, the vehicle can become unstable when operating in off-nominal and unknown conditions. On the contrary, model reference adaptive control (MRAC) is a nonlinear control technique which is inherently robust in the presence of parametric and unmatched uncertainties [75, Ch. 9]. This allows for robustness to carrying the sensor of potentially unknown mass and robustness to changes in the inertial properties of the overall system.

In the previous chapter, the switched version of the MRAC control law proved successful when mounting a camera to a wall. The moisture measurement scenario is almost identical to the wall mounting scenario in that we must push horizontally onto the wall to hold the sensor for measurements. As stated in Assumption 6.5.2, we assume that the wall is very close to vertical. Since we are using a standard quadrotor for our main vehicle, the system is underactuated, and a boom can be mounted slightly tilted to maintain an equilibrium while contacting the wall by being perpendicular to the gravity vector and acting though the vehicle center of gravity. Since the boom is fixed and not actuated, we have to assume that all mount points are vertical such that the wall contact orientation is the same for all occurrences. This quadrotor system utilizes four control inputs, $u_1(t), u_2(t), u_3(t), u_4(t)$. The first, $u_1(t)$, controls the total thrust force of the vehicle to control the altitude, and the other three control the attitude of the vehicle with respect to fixed axes attached to the vehicle measured relative to the ground. See Section 5.5.2 for modeling and control of a similar vehicle. By setting all of the tilt angles equal to zero, the dynamics simiplify to the vehicle used in this scenario.

Figure 6.7 provides a visual representation of how the switched control system works. The main point is the switching signal, which is user-defined, that activates the desired subsystem and control law. In this case, we assume the switching time is user-defined by

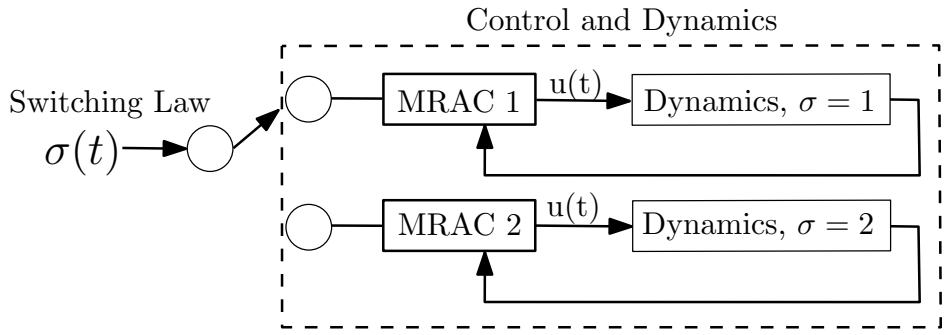

Figure 6.7: Structure of a switched MRAC law. The switching signal, $\sigma(\cdot)$, is a user-defined function of time which is piece-wise constant from the right and instantaneously changes at the times the vehicle contacts the wall. Each subsystem has its own dynamics and MRAC law, and the currently active subsystem provides the control vector, $u(t)$, to the vehicle.

approximating when we will hit the wall based on the reference signal. In reality, the exact switching instance is unknown, and to approximate it we can exploit the quadrotor dynamics. During the measurement phase, the vehicle is moving forwards, and at the point of contact, there is a sharp change in the local forward velocity from positive to 0. When this change is sensed through the accelerometer, the switching signal is changed to select the wall contact subsystem. Similarly, when the system detaches from the wall, there is a quick change in velocity from 0 to a negative value, and at this time, the switching signal is changed to activate the free flight subsystem. At these switching times, the current location of the vehicle is logged to a Comma Separated Values (CSV) file as explained in Section 6.5.6. The moisture measurement is recorded remotely through an app by the sensor manufacturer, and the moisture data and measurement locations are manually merged post-mission.

6.5.6 Data flow and BIM update

The overall data flow of the proposed system can be seen in Figure 6.8. Firstly, the native BIM model is exported as an STL file. Then, Algorithm 6 is used to convert the STL file to usable coordinates for the path planning algorithm. The path generating and optimization algorithm in Sections 6.5.2 and 6.5.3 is used to plan the path for the UAVs, and the mission is executed. The file generated on the vehicle during the mission contains the coordinates of

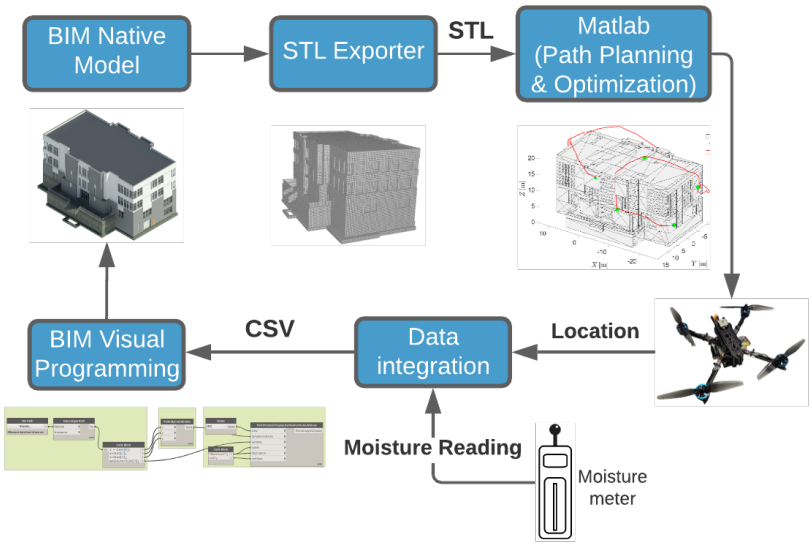

Figure 6.8: Data flow of the proposed system.

the measurement waypoints. For taking moisture readings, FlirTM MR59 moisture meter, as shown in Figure 6.9, is used. It is a pinless non-destructive type moisture meter that can be used on any surface type including concrete wall surfaces. The data is recorded remotely on a smartphone using the manufacturer's mobile application. The moisture data collected from the reading is recorded in an SQLite database. The SQLite database is exported from the storage of the smartphone to a computer. Then, the data is copied to a CSV file so that it can be merged with the coordinates data representing the location of the point the moisture reading has taken place. Once the data is integrated, a Dynamo script is used to convert the data into a format which can be visualized in the BIM model, which is then updated with the new measurements.

To visualize the data in the BIM model, Revit is used with Dynamo, which is a visual programming environment. Dynamo uses visual nodes to prepare scripts, instead of manual coding. It can be used to create and manipulate Revit elements. Primary advantage of using Dynamo is that it can be reused for multiple projects. Using Dynamo increases the power

of parametric design by interfacing with external data sources, such as CSV and web URLs [262].

Figure 6.9: The FlirTM MR59 moisture meter used in this work. To take measurements, the end of the sensor is placed against the wall and held in place.

Figure 6.10 shows the data flow in Dynamo. Data from the CSV file is read line-by-line. Figure 6.11 describes the Dynamo script developed to visualize the moisture data in the BIM model. *Data.ImportCSV* node in Dynamo is used to read CSV file. Coordinates read from the CSV file are stored in three arrays-x, y and z. Points are created in the model using the coordinates and the *Point.ByCoordinates* node. Here, it is important that the data follows the same coordinate system as the model. Therefore, the location data have to be transformed before generating the CSV file.

The in-built analysis node in Dynamo called *PointAnalysisDisplay.ByViewPointsandValues* is used to mark the points in the model and color it according to the moisture values recorded from the moisture meter. During this study, the colorscale shown in Figure 6.10 is used to visualize the moisture content. Deep blue represents high moisture content, while white represents low moisture content, see Figure 6.11.

The Dynamo script is independent of the model. This means it can be run on any BIM model in Revit as long as the data generated from the UAVs follows the same coordinate system as the model.

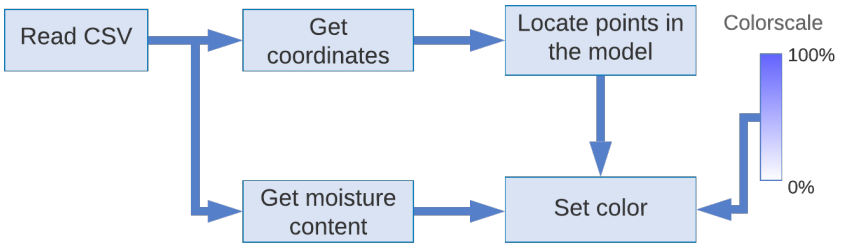

Figure 6.10: Process flow of visualizing measured data in BIM.

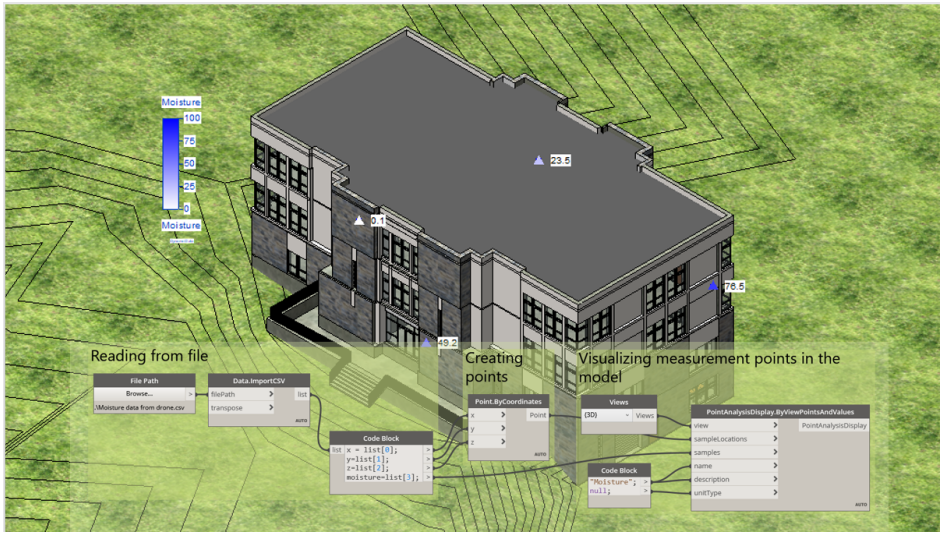

Figure 6.11: The Dynamo script.

6.5.7 Simulation environment and results

In this section, a full dynamic simulation environment is presented to test the data flow, path planner, and control law. To provide an accurate representation of the UAVs dynamics in free flight and in wall contact, a physics-based simulation environment is developed using Matlab's Simscape Multibody. A CAD model of the UAVs is created with similar mass and inertial properties to a prototype currently in development. To simulate a sensor, a sphere is added at the end of a boom which is mounted on the front of the vehicle. The sphere is chosen to be similar to the part of sensor that engages the wall, see Figure 6.9. To simulate the wall interactions between the vehicle and the building, the contact forces library is used,

and the contact is modeled as a sphere on a plane due to the shape of the end of the sensor. We ignore simulating the actual measurements taken by the sensor at each location. It is assumed that through 10s of wall contact with little to no movement suffices to ensure a successful moisture measurement.

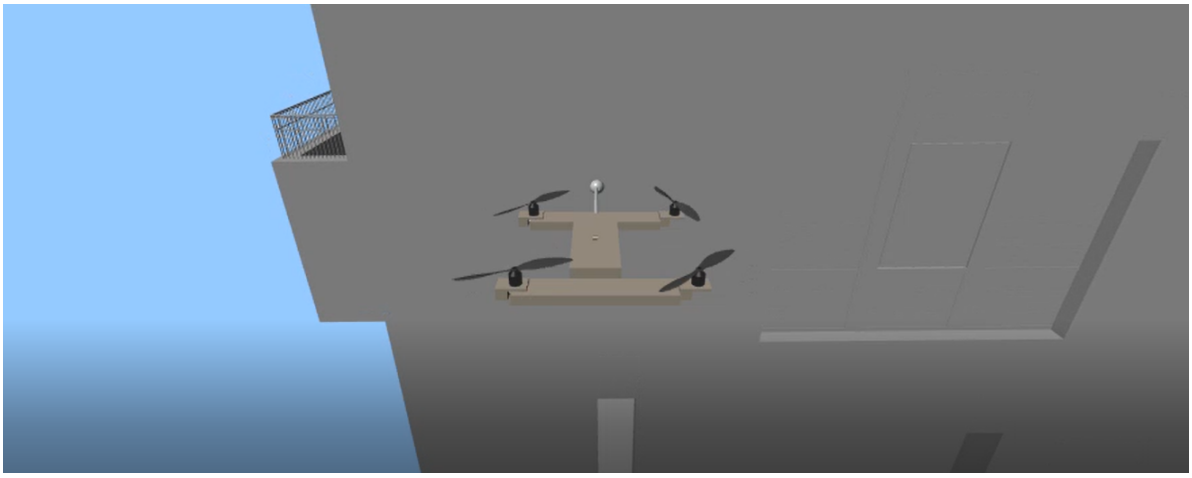

Figure 6.12: A screenshot of the Simscape Multibody simulation environment.

The dynamics of the UAVs are not calculated by the user, but are deduced by Simulink from the forces and moments induced on the vehicle using Newton's and Euler's laws. The user inputs are the angular velocities of the propellers which are calculated by the control law. These values are sent through a first order filter to simulate the realistic delay between commanded rates and realized rates. Lift and drag forces from the propellers are modeled as quadratic functions of the angular velocity with coefficients taken from manufacturers data [229]. The translational drag of the system is modeled as a function of the norm of the velocity with area estimated from the CAD model assuming very small pitch and roll angles, since the desired velocity of the mission is 1m/s.

The goal of the simulated mission is to simulate taking two autonomous measurements of user-defined locations on BFH with coordinates (x, y, z), $(14.7, 1.67, 11.0)$m and $(14.7, -6.33, 7.00)$m. This building, as in Figure 6.3, is four stories tall and the exterior walls of the top two floors consist of a precast concrete facade making it a good test case for our proposed framework. The available BIM is used to generate an STL file, which is

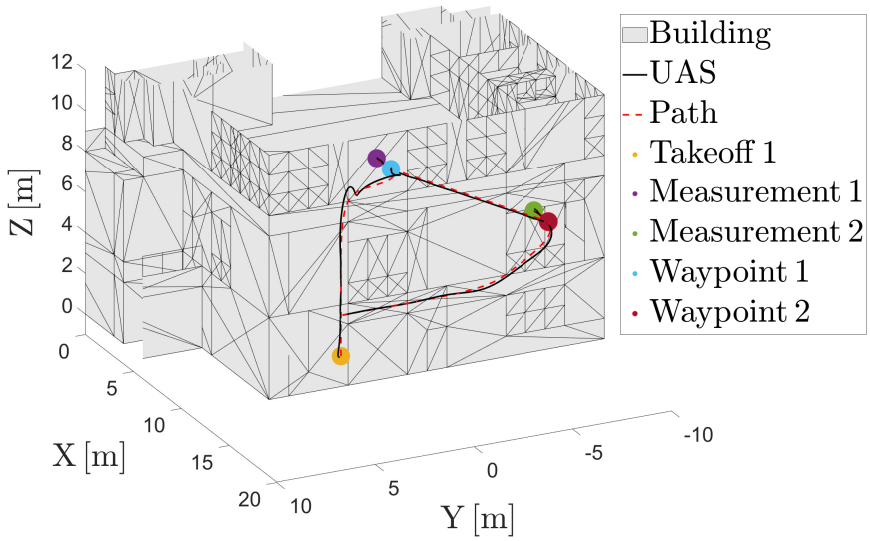

Figure 6.13: A plot showing a simulated mission wherein the vehicle is tasked with taking two different measurements on the building.

imported using Algorithm 6 and a user-defined grid size of 0.5m. Next, the path generator in Section 6.5.2 is run to generate the 3 possible paths. The heuristic, Algorithm 8 in Section 6.5.3 determines that the best path is to proceed from the start to the waypoint at $(14.7, 1.67, 11.0)$m, then proceed to $(14.7, -6.33, 7.00)$m, then return to the takeoff point, and the trajectory is saved to a text file exactly as it would be for standard missions.

In order to create a more realistic simulation scenario, noise and realistic update times are utilized. This vehicle is assumed to get its position and translational velocity via RTK measurements at a rate of 5Hz with error in the form white Gaussian noise with amplitude of 3cm and 5cm/s, respectively. The attitude and attitude rates of the vehicle, which are typically deduced by integrating data from accelerometers and gyroscopes, are fed back to the controller at a rate of 250Hz also with error in the form white Gaussian noise with amplitude of 0.02rad and 0.05rad/s, respectively. To simulate unknowns in the system, the mass of the system, the length of the boom, and the mass of the sensor are underestimated by 10% in the control algorithm.

The total simulation time is 95s, and the simulator is run at a fixed step of 0.005s for

integration. From time 0s to 4s, the vehicle is in the takeoff state, where the user-defined takeoff height is set to 2m. Then the UAVs transitions to the waypoint navigation state where it flies to the point in front of the measurement location, time 4s to 16s. The vehicle proceeds to enter the sensor measurement state and the vehicle moves forward, contacts the wall for 10s, then moves backwards. The UAVs then enters the waypoint navigation state again to maneuver to the second mounting location. The sensor measurement state is entered again from time 65s to 75s, and the second measurement is taken. After the second measurement, there are no more measurements, and the vehicle returns to the takeoff location and lands. Figure 6.13 shows the flight path and waypoints in 3D. It can be seen in the figure that during the periods in which the UAVs is not in the process of taking a measurement, it stays at least 2m from the building as desired.

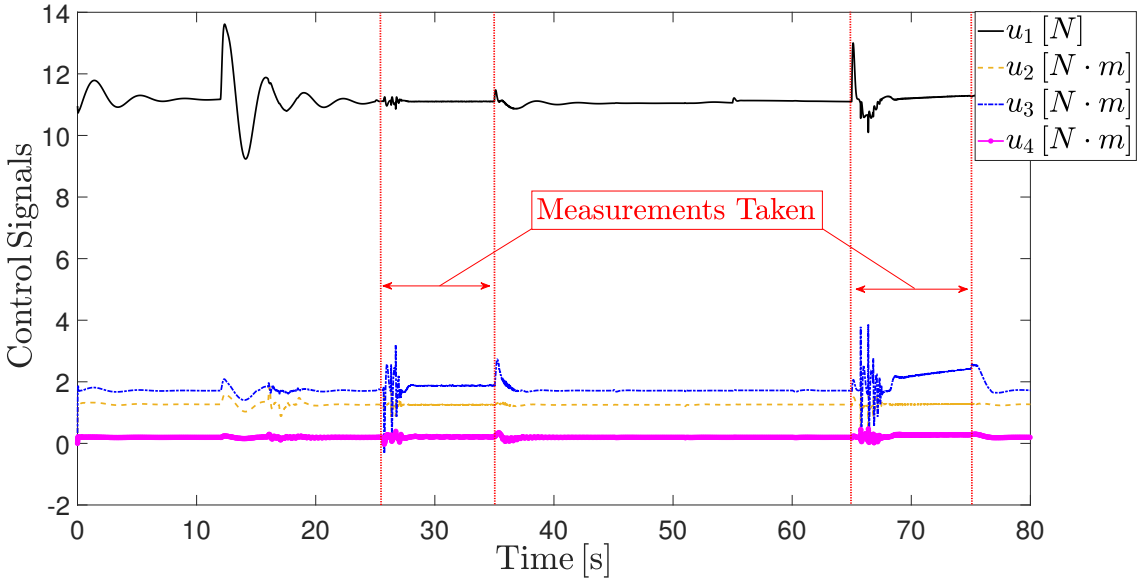

Figure 6.14: This plot shows the control inputs throughout the mission. It can be seen in the initial takeoff phase by the offset of $u_2(t)$ and $u_3(t)$, the roll and pitch controls, respectively, that the addition of the sensor causes the control law to work hard even in hover. The spikes in the control at time 25s and 65s are due to the wall contact disturbances which are well compensated by the adaptive control algorithm.

Figure 6.14 shows the control inputs to the vehicle throughout the mission. As can be seen from the figure, there are sharp oscillations which occur at times 25s and 65s. These

are the two times at which the UAVs comes into contact with the wall, and the most affected axis of the vehicle is the local $x(\cdot)$ axis, or the axis along the boom holding the measurement sensor. These strong discontinuities in the system dynamics are well compensated by the switched MRAC law, and accurate tracking is maintained. It is also interesting to note that $u_3(\cdot)$ is always biased to the positive axis. This is due to the extra moment on the vehicle from the simulated sensor being held in front of the vehicle. The sensor is also modeled as as asymmetric, and the $u_2(\cdot)$ control is also biased to account for the offset along the roll axis.

6.5.8 Prototype

A preliminary prototype vehicle has been constructed for potentially testing the algorithm developed in this work. The vehicle, as seen in Figure 6.15, is a standard quadrotor with an additional boom mounted on the front of the vehicle for carrying the measurement device and pressing against the wall. The boom prototype was 3D printed in-house using a polycarbonate (PC) filament with carbon fiber added to it. This filament was chosen due to its favorable strength to weight ratio and its rigidity. The model for the boom was designed in SolidWorks and extends 18cm from the front of the UAVs. The mounting platform at the end is at a 6 degree angle from vertical to compensate for the pitch of the vehicle while contacting the building. The boom was designed to withstand the lateral force that would be exerted from the building when contact is made. The printing orientation during 3D printing was optimized so the normal forces would not cause shear between the printing layers and cause it to break.

The power system of the vehicle consists of 4 T-Motor F60 Pro motors and 4 Aikon 20A electronic speed controllers (ESCs). The power supply is a 4 cell, 3000 mAh battery which yeilds an expected 20 minutes of flight time. According to the data from the manufacturers, the chosen motors are capable of producing a combined 6kg of thrust. The current vehicle

Figure 6.15: Prototype UAVs for testing path planning and control algorithms for contact-based inspection.

weight is 1.1kg without the moisture sensor, and this leaves a thrust to weight ratio of over 5 to 1, which exceeds the 2-1 ratio usually desired for steady flight. The selected moisture meter, pictured in Figure 6.9, weighs approximately 0.227kg, and mounting at the end of the boom would shift the center of gravity of the vehicle approximately 3cm. Taking this into account, it can be easily shown that the chosen motors are strong enough to overcome this limitation.

A standard mission would proceed as described in the following text. Firstly, the building STL file and the desired measurement locations would be provided. Next, the path generation and optimization algorithms which generates the path plan would be run offline. The path plan is then uploaded to the Odroid XU4 computer in a text file. The Odroid also manages the finite state machine governing the autopilot. The next desired waypoint is sent through a USB serial line to the on-board flight controller, a Pixhawk. The Pixhawk contains a gyroscope, an accelerometer, and an Extended Kalman Filter to deduce an accurate state

estimation by blending the sensors onboard as well as the localization data received through GPS or RTK. To control the vehicle, the control technique in Section 6.5.5 is coded in C++ and integrated into the PX4 flight stack on the Pixhawk. The Pixhawk directly actuates the motors by sending Pulse-width-modulation signals to the ESCs. While the algorithm is designed to be completely autonomous, for safety purposes the pilot can switch to a manual flight mode at any time during the mission to quickly and safely bring the UAVs back to the ground.

In this chapter, the work done in chapters 2-5 has been combined and implemented in a cohesive, real-world application wherein a method is created to rapidly utilize a BIM model to plan contact-based autonomous measurements of a building through the use of an unmanned aerial system. The path planner developed in Section 3.1.1 underlies the cost matrix used in the optimization model. The optimization model, where the optimal route for a UAV is found considering range constraints and the ability to swap batteries, is a special case of the one developed in Section 3.2. Due to the nature of the discontinuous contact forces incurred by taking the moisture measurements, the switched MRAC technique developed in chapter 5 is used to control the vehicle and ensure safe mission completion. To merge the measurement results back into the BIM model, a Dynamo script is developed to convert the measurement data to a format which can be seamlessly integrated into the BIM. A full simulation environment is developed to test the algorithms, and a preliminary prototype vehicle has been built and discussed.

Algorithm 6: This algorithm converts the STL file into an occupancy grid, $\mathcal{M}$, with user-defined resolution.

Result: An occupancy map, $\mathcal{M}$, which is representative of the modeled environment.

```
ρ_res = 0.5; % Resolution in [m]
ρ̃_res = 1/ρ_res;
free_space = 5; % [m]
Obstacle_list = [];
% Read the raw STL file
map_raw = Read_stl(map.stl);
% Add extra space around building for extra flight
map = insert_free_space(map_raw, free_space);
for All i ∈ n_p do
    % round to nearest resolution grid point
    p_i = round(ceil(p_i * ρ̃_res) /ρ̃_res,2);
    % Add vertices to obstacle list
    Obstacle_list = [Obstacle_list; p_i];
end
% Iterate connectivity list for interior points
for All a ∈ n_c do
    [i,j,k] = C(a,:);
    if p_i ≠ p_j, p_k & p_j ≠ p_k then
        Test_grid = make_grid(p_i, p_j, p_k);
        for All points in Test_grid do
            % Equations 1-4
            Result = Point_in_Triangle(p_i, p_j, p_k, p_0);
            if Result == 1 then
                Obstacle_list = [Obstacle_list; p_0];
            end
        end
    end
end
```

Algorithm 7: Construct_solution(cost,measurement_locations, refuel)

Result: Initial Solution.

```
route = {0}; % Initialize route at home location
unvisited = {1, 2, ··· , measurement_locations};
for i=1:num_measurement_locations do
    route = add_nearest_unvisited(route, unvisited, cost);
    battery_time = 0;
    for j=1:length(route) do
        battery_time + = cost(route(j-1),route(j));
        if battery_time > max_range then
            route = add_cheapest_refuel(route);
        end
        % If at battery swap, reset
        if route(j) ∈ N_b then
            battery_time = 0;
        end
    end
    update_unvisited_list(route, unvisited);
end
```

Algorithm 8: Local_search(Route, cost)

Result: Path plan.

```
route_star = route; c_star = cost(route_star);
while not_improved ¡ max_iterations do
    operation = randint(1,2);
    % k-opt
    if operation == 1 then
        k = randint(1,3);
        route_temp = location_swap(route, k);
        route_temp = ensure_feasibility(route_temp);
    end
    % Remove refuel
    if operation == 2 then
        route_temp = remove_refuel(route_temp);
    end
    c = cost(route_temp);
    if c < c_star & feasible(route_temp) == 1 then
        route_star = route_temp;
        c_star = c;
    end
end
```

Chapter 7

Conclusion and future research

7.1 Conclusion

In this dissertation, a method has been developed to address the routing and control problem for UAVs. Firstly, we studied the VRP and the SVRP to determine the effectiveness of using the SVRP, and the results of using multiple different objective functions were compared. Secondly, the SVRP formulation has been extended to a formulation more suitable for UAVs. The cost matrix for this formulation is built by generating path plans using an $A*$ algorithm which maintains a safe distance from obstacles, and the times to traverse these paths are adjusted for wind. Thirdly, two novel adaptive control techniques have been developed to ensure stability of the UAVs during mission scenarios ranging from delivering payload to installing payloads on a wall. Lastly, the previous results have been combined and implemented in a novel application where UAVs are used to take measurements of precast concrete and improve safety on construction sites.

In Chapter 2, a study was performed on both the classical VRP and the SVRP to determine the effects of using two different objective functions, namely the minimum cost and minimum last delivery functions. By performing a worst-case analysis, several relationships were developed between the two different formulations for the different objective functions. To test some of these relationships in numerical simulations on benchmark datasets, a heuristic has been developed based on embedding a random variable neighborhood search within an iterative local search framework. Numerical results suggest potentially lower costs for the

SVRP over the VRP when minimizing cost. It was also shown that using the minimum cost objective function can cause the last delivery time to increase significantly, doubled in some cases, and it was discussed that this can be a very poor choice when designing algorithms for disaster relief efforts.

In Chapter 3, an SVRP formulation was developed that is more suitable to UAVs than classical techniques presented in the literature. Firstly, a path planning technique using an A* search algorithm with a custom heuristic was developed to generate path plans to traverse between locations, whereas many classical techniques assume euclidean distances which can be unrealistic for UAVs flying in tighter areas near buildings. The flight times along these paths have been adjusted to account for a constant, known wind, since UAVs move with velocity relative to wind and not the ground. A power model for UAVs based on momentum theory was developed to establish a relationship between payload and power, hence energy, to explicitly account for the payloads effect on vehicle range in the optimization model. The full optimization model has been presented which accounts for visiting a set of known locations with delivery demands, additional demands may include surveillance times for taking images or mapping, the vehicles may swap batteries at known locations, and the power modeling is captured as a set of linear constraints. A realistic scenario was presented and discussed to validate the modeling approach.

Chapters 2 and 3 developed the framework for generating the optimal set of paths for the UAVs to follow, the next two chapters have been devoted to development of control laws that ensure the UAVs can successfully traverse these paths in the presence of uncertainties. Specifically, in Chapter 4 a novel MRAC law has been developed to guarantee a priori user-defined constraints on the trajectory tracking error and the control input. Both of these constraints are particularly applicable to UAVs since UAVs flying near buildings or persons should have safety guarantees. Furthermore, this control law improves upon one of the main drawbacks of classical MRAC laws, namely potentially poor transient performance. By utilizing a novel 2-layer MRAC framework, a user-defined rate of convergence can be set

guaranteeing that the system converges to the desired reference model before the transient dynamics has decayed. Numerical simulations have been performed on the roll dynamics of a delta-wing aircraft and the proposed control law has been shown to meet all design parameters, even in the presence of parametric and matched uncertainty. This control law is desirable for UAVs performing payload deliveries in environments where a priori tracking error guarantees are useful.

Next, we provided a novel control law that can be applied to UAVs that have to perform contact-based tasks. Specifically, in Chapter 5, an MRAC law for switched dynamical systems has been developed and mathematically proven in two frameworks, by analyzing the Carathéodory and Filippov solutions of discontinuous differential equations. This control law was applied to a thrust-vectoring quadcopter tasked with mounting a camera of unknown inertial properties to a wall. Experiments verified the usefulness of the proposed method.

Finally, after developing a solution method to both the routing and control problems for UAVs, a novel application has been developed wherein a UAV is tasked with taking several autonomous contact-based moisture measurements to both rapidly update the building information model and improve workers safety by removing the need to work at height. Using the ideas of path planning and optimal routing with battery swaps in Chapter 3, an optimization approach is developed for finding the best route. To solve the optimization problem quickly, a simple heuristic based on a k-opt procedure and random insertion and removal of battery swaps finds a fast solution to the problem. To ensure successful contact-based measurements, the switched MRAC law from Chapter 5 is implemented on the vehicle. Once the measurements are taken, a Dynamo script used with Revit generates and updates nodes within the BIM to track the moisture measurements. A physics-based Matlab simulator has been created, and a full simulation has been run where a quadrotor carrying a sensor on a boom successfully takes 2 autonomous moisture measurements. Finally, a preliminary prototype vehicle has been designed and built, and the process to perform a mission has been presented.

7.2 Recommendations for potential future research

In Chapter 2, the effects of choosing either the minimum cost or minimum last delivery objective function were studied. In future work, it may be desirable to also look at the effects of using the average delivery time, which could be considered as a customer service type of metric. In addition to a new objective function, one of the assumptions used in our work is that all vehicles should be used, and worst case-scenarios are commonly generated by adding nodes close to the depot to absorb extra vehicles that occur in the split delivery formulation. This is not necessarily applicable to real-world scenarios where vehicles that are unnecessary, or only increase the cost function, would not be used. Relaxing this assumption and performing more simulations could provide valuable insight into the effects.

The work in Chapter 3 focused on developing a model for an SVRP that is customized to UAVs. The current strategy is to generate path plans using an A* search, and assuming constant velocity, calculate the time it takes to traverse these paths. Since the goal of the overall routing problem is typically to either minimize the total distance traveled, or energy, or minimize the time for visiting the last location, an optimization approach that explicitly minimizes these values while taking vehicle dynamics into account would be worthwhile. The challenges become accurately modeling the constraints of obstacle avoidance and making accurate dynamical models. The model of power used in the current optimization model is based on simple momentum theory and has been shown to be inaccurate in some scenarios, specifically when the loading on the rotors increase at larger diameters [263]. Adding blade element theory to this model can improve accuracy, but most likely the optimization problem becomes a nonlinear program instead of a linear one, which may or may not be solvable by available solvers. While solvers can sometimes find the optimal solutions, it would be desirable to implement a heuristic method to create fast approximately optimal solutions to the formulation, similar to the heuristic in Chapter 2. Some authors have developed a heuristic for UAV routing where the range depends on the carried payload based on simulated

annealing [52], but this formulation considers only homogeneous fleets, the travel times are symmetric, and the service times are fixed unlike our case where services can be optimization variables.

While the work in Chapter 4 has extended the state of the art in MRAC, the current framework does not guarantee any convergence of the adaptive gains. Future work directions could involve extending the proposed direct MRAC laws for prescribed performance to estimate the unknown plant parameters. These mixed direct-indirect adaptive controls will be produced extending the concurrent learning MRAC and the composite learning MRAC frameworks already available in the literature.

The field of adaptive control for switched dynamical systems is fairly new relative to adaptive control in general. Our control law in Chapter 5 is the first MRAC law for switched, unknown nonlinear systems, but the assumption that the switching law is a known function of time can be a limiting factor in some applications. Consider the tilt-rotor system that mounted the camera to the wall. While in a controlled environment, the switching times may be known quite accurately, in a more dynamic environment where the vehicle position is known with large uncertainty, or the contact surface is moving, the switching time may be unknown. Future work for the switched MRAC includes developing an integrated switching law and adaptive control scheme where the adaptive controller switches based on the states of the system or according to some higher level logic-based control law. Similar work has been done for switched affine systems [264], but it seems a state-dependent switching for nonlinear, unknown systems with an MRAC law is still unexplored.

The combined application of the work in this dissertation presented in Chapter 6 culminated in successful numerical simulations. Future research includes fully integrated testing to experimentally verify the results of the work. Additionally, the some of the restrictions of this chapter could be addressed. While the current algorithm is restrictive to a known and static environment, the system could be upgraded with cameras and a Simultaneous

Localization and Mapping (SLAM) algorithm which could both update the environment in real-time and update the vehicle's location within the map without the need of GPS. Furthermore, to potentially address the problem of taking measurements of slanted surfaces, additional actuation could be explored on the vehicle, either through additional degrees of freedom of the arm, or through actuating the propeller thrust in a tilt-rotor format, such as the one in Section 5.5.

Bibliography

[1] M. M. Silva, A. Subramanian, and L. S. Ochi, "An iterated local search heuristic for the split delivery vehicle routing problem," *Computers & Operations Research*, vol. 53, pp. 234–249, 2015.

[2] J. Fisher. (2019) First consumer drone delivery service takes off in virginia. [Online]. Available: https://www.fleetowner.com/technology/article/21704392/first-consumer-drone-delivery-service-takes-off-in-virginia

[3] R. Beard, D. Kingston, M. Quigley, D. Snyder, R. Christiansen, W. Johnson, T. McLain, and M. Goodrich, "Autonomous vehicle technologies for small fixed-wing UAVs," *Journal of Aerospace Computing, Information, and Communication*, vol. 2, no. 1, pp. 92–108, 2005.

[4] U. Ozdemir, Y. O. Aktas, A. Vuruskan, Y. Dereli, A. F. Tarhan, K. Demirbag, A. Erdem, G. D. Kalaycioglu, I. Ozkol, and G. Inalhan, "Design of a commercial hybrid vtol UAV system," *Journal of Intelligent & Robotic Systems*, vol. 74, no. 1, pp. 371–393, 2014.

[5] A. S. Saeed, A. B. Younes, S. Islam, J. Dias, L. Seneviratne, and G. Cai, "A review on the platform design, dynamic modeling and control of hybrid UAVs," in *2015 International Conference on Unmanned Aircraft Systems*. IEEE, 2015, pp. 806–815.

[6] U. Papa, "Introduction to unmanned aircraft systems (UAS)," in *Embedded Platforms for UAS Landing Path and Obstacle Detection*. Springer, 2018, pp. 1–11.

[7] D. Glade, "Unmanned aerial vehicles: Implications for military operations," Air Univ Press Maxwell Afb Al, Tech. Rep., 2000.

[8] P. Ramesh and J. M. L. Jeyan, "Comparative analysis of the impact of operating parameters on military and civil applications of mini unmanned aerial vehicle (UAV)," in *AIP Conference Proceedings*, vol. 2311, no. 1. AIP Publishing LLC, 2020.

[9] P. Doherty and P. Rudol, "A UAV search and rescue scenario with human body detection and geolocalization," in *Australasian Joint Conference on Artificial Intelligence*. Springer, 2007, pp. 1–13.

[10] T. Tomic, K. Schmid, P. Lutz, A. Domel, M. Kassecker, E. Mair, I. L. Grixa, F. Ruess, M. Suppa, and D. Burschka, "Toward a fully autonomous UAV: Research platform for indoor and outdoor urban search and rescue," *IEEE robotics & automation magazine*, vol. 19, no. 3, pp. 46–56, 2012.

[11] M. A. Goodrich, B. S. Morse, D. Gerhardt, J. L. Cooper, M. Quigley, J. A. Adams, and C. Humphrey, "Supporting wilderness search and rescue using a camera-equipped mini UAV," *Journal of Field Robotics*, vol. 25, no. 1-2, pp. 89–110, 2008.

[12] D. Hausamann, W. Zirnig, G. Schreier, and P. Strobl, "Monitoring of gas pipelines–a civil UAV application," *Aircraft Engineering and Aerospace Technology*, 2005.

[13] M. Erdelj, E. Natalizio, K. R. Chowdhury, and I. F. Akyildiz, "Help from the sky: Leveraging UAVs for disaster management," *IEEE Pervasive Computing*, vol. 16, no. 1, pp. 24–32, 2017.

[14] M. Erdelj and E. Natalizio, "UAV-assisted disaster management: Applications and open issues," in *international conference on computing, networking and communications*. IEEE, 2016, pp. 1–5.

[15] D. W. Casbeer, R. W. Beard, T. W. McLain, S.-M. Li, and R. K. Mehra, "Forest fire monitoring with multiple small UAVs," in *Proceedings of the American Control Conference*. IEEE, 2005, pp. 3530–3535.

[16] C. Yuan, Y. Zhang, and Z. Liu, "A survey on technologies for automatic forest fire monitoring, detection, and fighting using unmanned aerial vehicles and remote sensing techniques," *Canadian journal of forest research*, vol. 45, no. 7, pp. 783–792, 2015.

[17] N. Anwar, M. A. Izhar, and F. A. Najam, "Construction monitoring and reporting using drones and unmanned aerial vehicles (UAVs)," in *The Tenth International Conference on Construction in the 21st Century*, 2018, pp. 2–4.

[18] J. J. Lin, K. K. Han, and M. Golparvar-Fard, "A framework for model-driven acquisition and analytics of visual data using UAVs for automated construction progress monitoring," in *Computing in Civil Engineering 2015*, 2015, pp. 156–164.

[19] P. Toth and D. Vigo, *The vehicle routing problem*. SIAM, 2002.

[20] R. B. Anderson, J.-P. Burke, J. A. Marshall, and A. L'Afflitto, "Robust adaptive control for constrained tilt-rotor quadcopters of unknown inertial properties," in *American Control Conference*, 2019, pp. 2922–2927.

[21] P. E. Pounds, D. R. Bersak, and A. M. Dollar, "Stability of small-scale UAV helicopters and quadrotors with added payload mass under PID control," *Autonomous Robots*, vol. 33, no. 1, pp. 129–142, 2012.

[22] A. L. Salih, M. Moghavvemi, H. A. Mohamed, and K. S. Gaeid, "Flight PID controller design for a UAV quadrotor," *Scientific research and essays*, vol. 5, no. 23, pp. 3660–3667, 2010.

[23] A. M. Campbell, D. Vandenbussche, and W. Hermann, "Routing for relief efforts," *Transportation science*, vol. 42, no. 2, pp. 127–145, 2008.

[24] Z. Xu, L. Xu, and C.-L. Li, "Approximation results for min-max path cover problems in vehicle routing," *Naval Research Logistics (NRL)*, vol. 57, no. 8, pp. 728–748, 2010.

[25] J.-F. Cordeau and M. Maischberger, "A parallel iterated tabu search heuristic for vehicle routing problems," *Computers & Operations Research*, vol. 39, no. 9, pp. 2033–2050, 2012.

[26] B. M. Baker and M. Ayechew, "A genetic algorithm for the vehicle routing problem," *Computers & Operations Research*, vol. 30, no. 5, pp. 787–800, 2003.

[27] J. Berger and M. Barkaoui, "A new hybrid genetic algorithm for the capacitated vehicle routing problem," *Journal of the operational Research Society*, vol. 54, no. 12, pp. 1254–1262, 2003.

[28] A. S. Alfa, S. S. Heragu, and M. Chen, "A 3-opt based simulated annealing algorithm for vehicle routing problems," *Computers & Industrial Engineering*, vol. 21, no. 1-4, pp. 635–639, 1991.

[29] J. E. Bell and P. R. McMullen, "Ant colony optimization techniques for the vehicle routing problem," *Advanced engineering informatics*, vol. 18, no. 1, pp. 41–48, 2004.

[30] J. F. Sze, S. Salhi, and N. Wassan, "The cumulative capacitated vehicle routing problem with min-sum and min-max objectives: An effective hybridisation of adaptive variable neighbourhood search and large neighbourhood search," *Transportation Research Part B: Methodological*, vol. 101, pp. 162–184, 2017.

[31] C. Ren, "Solving min-max vehicle routing problem." *Journal of Software*, vol. 6, no. 9, pp. 1851–1856, 2011.

[32] B. L. Golden, G. Laporte, and É. D. Taillard, "An adaptive memory heuristic for a class of vehicle routing problems with minmax objective," *Computers & Operations Research*, vol. 24, no. 5, pp. 445–452, 1997.

[33] E. Yakıcı and O. Karasakal, "A min–max vehicle routing problem with split delivery and heterogeneous demand," *Optimization Letters*, vol. 7, no. 7, pp. 1611–1625, 2013.

[34] C. Archetti, M. G. Speranza, and A. Hertz, "A tabu search algorithm for the split delivery vehicle routing problem," *Transportation science*, vol. 40, no. 1, pp. 64–73, 2006.

[35] R. E. Aleman and R. R. Hill, "A tabu search with vocabulary building approach for the vehicle routing problem with split demands," *International Journal of Metaheuristics*, vol. 1, no. 1, pp. 55–80, 2010.

[36] C. Archetti, M. G. Speranza, and M. W. Savelsbergh, "An optimization-based heuristic for the split delivery vehicle routing problem," *Transportation Science*, vol. 42, no. 1, pp. 22–31, 2008.

[37] M. Boudia, C. Prins, and M. Reghioui, "An effective memetic algorithm with population management for the split delivery vehicle routing problem," in *International Workshop on Hybrid Metaheuristics*. Springer, 2007, pp. 16–30.

[38] J. H. Wilck IV and T. M. Cavalier, "A genetic algorithm for the split delivery vehicle routing problem," *American Journal of Operations Research*, vol. 2, no. 2, 2012.

[39] P. H. V. Penna, A. Subramanian, and L. S. Ochi, "An iterated local search heuristic for the heterogeneous fleet vehicle routing problem," *Journal of Heuristics*, vol. 19, no. 2, pp. 201–232, 2013.

[40] C. Archetti, M. W. P. Savelsbergh, and M. G. Speranza, "Worst-case analysis for split delivery vehicle routing problems," *Transportation Science*, vol. 40, no. 2, pp. 226–234, 2006.

[41] A. M. Campbell, D. Vandenbussche, and W. Hermann, "Routing for relief efforts," *Transportation Science*, vol. 42, no. 2, pp. 127–145, 2008.

[42] P. Toth and D. Vigo, "Branch-and-bound algorithms for the capacitated VRP," in *The vehicle routing problem*. SIAM, 2002, pp. 29–51.

[43] J. H. Wilck IV and T. M. Cavalier, "A construction heuristic for the split delivery vehicle routing problem," *American Journal of Operations Research*, vol. 2, no. 2, 2012.

[44] J.-M. Belenguer, M. Martinez, and E. Mota, "A lower bound for the split delivery vehicle routing problem," *Operations research*, vol. 48, no. 5, pp. 801–810, 2000.

[45] G. Reinelt, "Tsplib—a traveling salesman problem library," *ORSA journal on computing*, vol. 3, no. 4, pp. 376–384, 1991.

[46] P. Augerat, D. Naddef, J. Belenguer, E. Benavent, A. Corberan, and G. Rinaldi, "Computational results with a branch and cut code for the capacitated vehicle routing problem," Institut National Polytechnique, Tech. Rep., 1995.

[47] S. Chen, B. Golden, and E. Wasil, "The split delivery vehicle routing problem: Applications, algorithms, test problems, and computational results," *Networks: An International Journal*, vol. 49, no. 4, pp. 318–329, 2007.

[48] M. Dror and P. Trudeau, "Savings by split delivery routing," *Transportation Science*, vol. 23, no. 2, pp. 141–145, 1989.

[49] M. Foltyński, "Electric fleets in urban logistics," *Procedia-Social and Behavioral Sciences*, vol. 151, pp. 48–59, 2014.

[50] S. Erdoğan and E. Miller-Hooks, "A green vehicle routing problem," *Transportation research part E: logistics and transportation review*, vol. 48, no. 1, pp. 100–114, 2012.

[51] G. Hiermann, J. Puchinger, S. Ropke, and R. F. Hartl, "The electric fleet size and mix vehicle routing problem with time windows and recharging stations," *European Journal of Operational Research*, vol. 252, no. 3, pp. 995–1018, 2016.

[52] K. Dorling, J. Heinrichs, G. G. Messier, and S. Magierowski, "Vehicle routing problems

for drone delivery," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 1, pp. 70–85, 2016.

[53] J. Greblicki and M. Walczyński, "Determination of the optimal routes for autonomous unmanned aerial vehicle under varying wind with using of the traveling salesman problem algorithm," in *IFIP International Conference on Computer Information Systems and Industrial Management*. Springer, 2016, pp. 334–341.

[54] A. Kundu and T. I. Matis, "A delivery time reduction heuristic using drones under windy conditions," in *IIE Annual Conference. Proceedings*. Institute of Industrial and Systems Engineers, 2017, pp. 1864–1869.

[55] L. De Filippis and G. Guglieri, *Advanced Graph Search Algorithms for Path Planning of Flight Vehicles*. Intech, 2012, pp. 157–192.

[56] S. Primatesta, G. Guglieri, and A. Rizzo, "A risk-aware path planning strategy for UAVs in urban environments," *Journal of Intelligent & Robotic Systems*, vol. 95, no. 2, pp. 629–643, 2019.

[57] S. Koenig and M. Likhachev, "Fast replanning for navigation in unknown terrain," *Transactions on Robotics*, vol. 21, no. 3, pp. 354–363, 2005.

[58] A. A. Maw, M. Tyan, and J.-W. Lee, "iADA*: Improved anytime path planning and replanning algorithm for autonomous vehicle," *Journal of Intelligent & Robotic Systems*, vol. 100, no. 3, pp. 1005–1013, 2020.

[59] R. Bohlin and L. E. Kavraki, "Path planning using lazy PRM," in *International Conference on Robotics and Automation*, vol. 1. Paris, France: IEEE, 2000, pp. 521–528.

[60] A. Madridano, A. Al-Kaff, and D. Martin, "3D trajectory planning method for UAVs swarm in building emergencies," *Sensors*, vol. 20, no. 3, p. 642, 2020.

[61] M. Kothari and I. Postlethwaite, "A probabilistically robust path planning algorithm for UAVs using rapidly-exploring random trees," *Journal of Intelligent & Robotic Systems*, vol. 71, no. 2, pp. 231–253, 2013.

[62] I. Noreen, A. Khan, H. Ryu, N. L. Doh, and Z. Habib, "Optimal path planning in cluttered environment using RRT*-AB," *Intelligent Service Robotics*, vol. 11, no. 1, pp. 41–52, 2018.

[63] E. Masehian and M. Amin-Naseri, "A Voronoi diagram-visibility graph-potential field compound algorithm for robot path planning," *Journal of Robotic Systems*, vol. 21, no. 6, pp. 275–300, 2004.

[64] J. Marshall, R. B. Anderson, W. Chen, E. N. Johnson, and A. L'Afflitto, "A guidance system for tactical autonomous unmanned aerial vehicles," *Journal of Intelligent & Robotic Systems*, 2020.

[65] J.-C. Latombe, *Robot motion planning*. Springer Science & Business Media, 2012, vol. 124.

[66] B. Whitehead and S. Bieniawski, "Model reference adaptive control of a quadrotor UAV," in *AIAA Guidance, Navigation, and Control Conference*, 2010, p. 8148.

[67] A. J. Chorin, J. E. Marsden, and J. E. Marsden, *A mathematical introduction to fluid mechanics*. Springer, 1990, vol. 168.

[68] J. M. Seddon and S. Newman, *Basic helicopter aerodynamics*. John Wiley & Sons, 2011, vol. 40.

[69] I. Motawa and A. Kardakou, "Unmanned aerial vehicles (UAVs) for inspection in construction and building industry," in *International Operation & Maintenance Conference*, 2018.

[70] H. Freimuth, J. Müller, and M. König, "Simulating and executing UAV-assisted inspections on construction sites," in *International Symposium on Automation and Robotics in Construction*, 2017.

[71] T. Rakha and A. Gorodetsky, "Review of unmanned aerial system (UAS) applications in the built environment: Towards automated building inspection procedures using drones," *Automation in Construction*, vol. 93, pp. 252–264, 2018.

[72] C. E. Miller, A. W. Tucker, and R. A. Zemlin, "Integer programming formulation of traveling salesman problems," *Journal of the ACM*, vol. 7, no. 4, pp. 326–329, 1960.

[73] R. B. Anderson, J. A. Marshall, and A. L'Afflitto, "Constrained robust model reference adaptive control of a tilt-rotor quadcopter pulling an unmodeled cart," *IEEE Transactions on Aerospace and Electronic Systems*, 2020.

[74] A. L'Afflitto, R. B. Anderson, and K. Mohammadi, "An introduction to nonlinear robust control for unmanned quadrotor aircraft," *IEEE Control Systems Magazine*, vol. 38, no. 3, pp. 102–121, 2018.

[75] E. Lavretsky and K. Wise, *Robust and Adaptive Control: With Aerospace Applications*. London, UK: Springer, 2012.

[76] B. Peterson and K. Narendra, "Bounded error adaptive control," *IEEE Transactions on Automatic Control*, vol. 27, no. 6, pp. 1161–1168, 1982.

[77] P. Ioannou and P. Kokotovic, *Adaptive Systems with Reduced Models*. New York, NY: Springer, 1983.

[78] K. Narendra and A. Annaswamy, "A new adaptive law for robust adaptation without persistent excitation," *IEEE Transactions on Automatic Control*, vol. 32, no. 2, pp. 134–145, 1987.

[79] G. Navarro-Guerrero and Y. Tang, "Fractional order model reference adaptive control for anesthesia," *International Journal of Adaptive Control and Signal Processing*, vol. 31, no. 9, pp. 1350–1360, 2017.

[80] C. Ma, J. Lam, and F. L. Lewis, "Trajectory regulating model reference adaptive controller for robotic systems," *IEEE Transactions on Control Systems Technology*, vol. 27, no. 6, pp. 2749–2756, 2019.

[81] S. A. Davari and J. Rodriguez, "Predictive direct voltage control of induction motor with mechanical model consideration for sensorless applications," *IEEE Journal of Emerging and Selected Topics in Power Electronics*, vol. 6, no. 4, pp. 1990–2000, 2018.

[82] E. Arabi, B. C. Gruenwald, T. Yucelen, and N. T. Nguyen, "A set-theoretic model reference adaptive control architecture for disturbance rejection and uncertainty suppression with strict performance guarantees," *International Journal of Control*, vol. 91, no. 5, pp. 1195–1208, 2018.

[83] A. L'Afflitto, "Barrier Lyapunov functions and constrained model reference adaptive control," *IEEE Control Systems Letters*, vol. 2, no. 3, pp. 441–446, 2018.

[84] J. Na, G. Herrmann, and K. Zhang, "Improving transient performance of adaptive control via a modified reference model and novel adaptation," *International Journal of Robust and Nonlinear Control*, vol. 27, no. 8, pp. 1351–1372, 2017.

[85] B. Mirkin and P.-O. Gutman, "Tube model reference adaptive control," *Automatica*, vol. 49, no. 4, pp. 1012–1018, 2013.

[86] M. C. Turner, J. Sofrony, and E. Prempain, "Anti-windup for model-reference adaptive control schemes with rate-limits," *Systems & Control Letters*, vol. 137, p. 104630, 2020.

[87] C. P. Bechlioulis and G. A. Rovithakis, "Robust adaptive control of feedback lineariz-

able MIMO nonlinear systems with prescribed performance," *IEEE Transactions on Automatic Control*, vol. 53, no. 9, pp. 2090–2099, 2008.

[88] C. P. Bechlioulis and G. A. Rovithakis, "Adaptive control with guaranteed transient and steady state tracking error bounds for strict feedback systems," *Automatica*, vol. 45, no. 2, pp. 532 – 538, 2009.

[89] K. Narendra and A. Annaswamy, *Stable Adaptive Systems*. Englewood Cliffs, NJ: Dover, 2012.

[90] G. Chowdhary and E. N. Johnson, "Concurrent learning for convergence in adaptive control without persistency of excitation," in *IEEE Conference on Decision and Control*, 2010, pp. 3674–3679.

[91] G. Chowdhary, T. Yucelen, M. Mühlegg, and E. N. Johnson, "Concurrent learning adaptive control of linear systems with exponentially convergent bounds," *International Journal of Adaptive Control and Signal Processing*, vol. 27, no. 4, pp. 280–301, 2013.

[92] G. Chowdhary, M. Mühlegg, and E. N. Johnson, "Exponential parameter and tracking error convergence guarantees for adaptive controllers without persistency of excitation," *International Journal of Control*, vol. 87, no. 8, pp. 1583–1603, 2014.

[93] M. A. Duarte and K. S. Narendra, "Combined direct and indirect approach to adaptive control," *IEEE Transactions on Automatic Control*, vol. 34, no. 10, pp. 1071–1075, 1989.

[94] K. S. Narendra and M. A. Duarte, "Application of robust adaptive control using combined direct and indirect methods," *International Journal of Adaptive Control and Signal Processing*, vol. 3, no. 2, pp. 131–142, 1989.

[95] G. Kreisselmeier, "Adaptive observers with exponential rate of convergence," *IEEE Transactions on Automatic Control*, vol. 22, no. 1, pp. 2–8, 1977.

[96] N. T. Nguyen, *Model-Reference Adaptive Control*. London, UK: Springer, 2018.

[97] P. Ioannou and J. Sun, *Robust Adaptive Control*. Englewood Cliffs, NJ: Dover, 2012.

[98] T. Yucelen, G. De La Torre, and E. N. Johnson, "Improving transient performance of adaptive control architectures using frequency-limited system error dynamics," *International Journal of Control*, vol. 87, no. 11, pp. 2383–2397, 2014.

[99] T. E. Gibson, A. M. Annaswamy, and E. Lavretsky, "On adaptive control with closed-loop reference models: Transients, oscillations, and peaking," *IEEE Access*, vol. 1, pp. 703–717, 2013.

[100] J. Yang, Y. Liu, J. Na, and G. Gao, *Improving Transient Performance of Modified Model Reference Adaptive Control*. Singapore: Springer, 2018, pp. 331–343.

[101] N. T. Nguyen, "Optimal control modification for robust adaptive control with large adaptive gain," *Systems & Control Letters*, vol. 61, no. 4, pp. 485–494, 2012.

[102] A. Datta and P. A. Ioannou, "Performance analysis and improvement in model reference adaptive control," *IEEE Transactions on Automatic Control*, vol. 39, no. 12, pp. 2370–2387, 1994.

[103] J. Yang, J. Na, and G. Gao, "Robust adaptive control with a modified controller for transient response improvement," in *International Conference on Modelling, Identification and Control*, 2017, pp. 929–934.

[104] Y. Yang, X. Chen, and C. Li, "Transient performance improvement in model reference adaptive control using H_∞ optimal method," *Journal of the Franklin Institute*, vol. 352, no. 1, pp. 16–32, 2015.

[105] J. F. Quindlen, G. Chowdhary, and J. P. How, "Hybrid model reference adaptive control for unmatched uncertainties," in *American Control Conference*, 2015, pp. 1125–1130.

[106] G. Joshi and G. Chowdhary, "Hybrid direct-indirect adaptive control of nonlinear system with unmatched uncertainty," in *International Conference on Control, Decision and Information Technologies*, 2019, pp. 127–132.

[107] G. V. Chowdhary and E. N. Johnson, "Theory and flight-test validation of a concurrent-learning adaptive controller," *Journal of Guidance, Control, and Dynamics*, vol. 34, no. 2, pp. 592–607, 2011.

[108] H. Wang and J. Sun, "Modified model reference adaptive control with saturated inputs," in *IEEE Conference on Decision and Control*, vol. 4, 1992, pp. 3255–3256.

[109] F. Z. Chaoui, F. Giri, L. Dugard, J. M. Dion, and M. M'saad, "Adaptive tracking with saturating input and controller integral action," *IEEE Transactions on Automatic Control*, vol. 43, no. 11, pp. 1638–1643, 1998.

[110] C. C. Palerm and B. W. Bequette, "Direct model reference adaptive control and saturation constraints," *IFAC Proceedings Volumes*, vol. 35, no. 1, pp. 409–414, 2002, iFAC World Congress.

[111] M. Kanamori and M. Tomizuka, "Model reference adaptive control of linear systems with input saturation," in *IEEE International Conference on Control Applications*, vol. 2, 2004, pp. 1318–1323.

[112] E. Lavretsky and N. Hovakimyan, "Stable adaptation in the presence of input constraints," *Systems & Control Letters*, vol. 56, no. 11, pp. 722–729, 2007.

[113] E. N. Johnson and A. J. Calise, "Neural network adaptive control of systems with input saturation," in *American Control Conference*, vol. 5, 2001, pp. 3527–3532.

[114] S. Kannan and E. N. Johnson, "Adaptive control with a nested saturation reference model," in *AIAA Guidance, Navigation, and Control Conference*, 2003, pp. 1–11.

[115] G. Kreisselmeier and K. Narendra, "Stable model reference adaptive control in the presence of bounded disturbances," *IEEE Transactions on Automatic Control*, vol. 27, no. 6, pp. 1169–1175, 1982.

[116] J. B. Pomet and L. Praly, "Adaptive nonlinear regulation: estimation from the Lyapunov equation," *IEEE Transactions on Automatic Control*, vol. 37, no. 6, pp. 729–740, 1992.

[117] W. M. Haddad and V. Chellaboina, *Nonlinear Dynamical Systems and Control: A Lyapunov-Based Approach*. Princeton, NJ: Princeton Univ. Press, 2008.

[118] S. Boyd, L. Ghaoui, E. Feron, and V. Balakrishnan, *Linear Matrix Inequalities in System and Control Theory*. Philadelphia, PA: SIAM, 1994.

[119] P. J. Antsaklis and A. N. Michel, *A Linear Systems Primer*. Boston, MA: Birkhäuser, 2007.

[120] B. Kågström, "Bounds and perturbation bounds for the matrix exponential," *IEEE Transactions on Biomedical Engineering*, vol. 17, no. 1, pp. 39–57, 2015.

[121] D. S. Bernstein, *Matrix Mathematics: Theory, Facts, and Formulas, Second Edition*. Princeton, NJ: Princeton University Press, 2009.

[122] M. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modeling and Control*. New York, NY: Wiley, 2005.

[123] A. Loria, E. Panteley, and M. Maghenem, "Strict Lyapunov functions for model reference adaptive control: Application to Lagrangian systems," *IEEE Transactions on Automatic Control*, vol. 64, no. 7, pp. 3040–3045, 2019.

[124] H. K. Khalil, *Nonlinear Systems*. Princeton, NJ: Prentice Hall, 2002.

[125] A. L'Afflitto, *A Mathematical Perspective on Flight Dynamics and Control*. London, UK: Springer, 2017.

[126] C. Alexander and M. Sadiku, *Fundamentals of Electric Circuits*. New York, NY: McGraw Hill, 2008.

[127] A. V. Oppenheim, W. Schaffer, Ronald, and J. R. Buck, *Discrete-time signal processing*. Upper Saddle River, NJ: Prentice Hall, 1999.

[128] A. Gelb, *Applied optimal estimation*. Cambridge, MA: MIT press, 1974.

[129] T. Homer, M. Mahmood, and P. Mhaskar, "A trajectory-based method for constructing null controllable regions," *International Journal of Robust and Nonlinear Control*, vol. 30, no. 2, pp. 776–786, 2020.

[130] L. Zaccarian and A. R. Teel, *Modern Anti-windup Synthesis: Control Augmentation for Actuator Saturation*. Princeton, NJ: Princeton University Press, 2011.

[131] E. Sontag and H. J. Sussmann, "Nonsmooth control-Lyapunov functions," in *IEEE Conference on Decision and Control*, vol. 3, 1995, pp. 2799–2805.

[132] T. Homer and P. Mhaskar, "Using null controllable regions to stabilize nonlinear systems," in *American Control Conference*, 2018, pp. 3141–3146.

[133] R. Vrabel, "Local null controllability of the control-affine nonlinear systems with time-varying disturbances," *European Journal of Control*, vol. 40, pp. 80–86, 2018.

[134] E. B. Lee and L. Markus, *Foundations of Optimal Control Theory*. Philadelphia, PA: SIAM, 1967.

[135] J. Na, Q. Chen, X. Ren, and Y. Guo, "Adaptive prescribed performance motion control of servo mechanisms with friction compensation," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 1, pp. 486–494, 2013.

[136] B. Brogliato, *Nonsmooth Mechanics: Models, Dynamics and Control*, ser. Communications and Control Engineering. London, UK: Springer, 1999.

[137] F. Pfeiffer and C. Glocker, *Multibody Dynamics with Unilateral Contacts*. Vienna, Austria: Springer, 2000.

[138] A. E. Bryson, *Applied Optimal Control: Optimization, Estimation and Control*. New York, NY: Taylor & Francis, 1975.

[139] C. Edwards and S. Spurgeon, *Sliding Mode Control: Theory And Applications*, ser. Series in Systems and Control. New York, NY: Taylor & Francis, 1998.

[140] A. Levant, "Introduction to high-order sliding modes," in *Sliding Mode Control in Engineering*, W. Perruquetti and J. P. Barbot, Eds. Basel, Switzerland: Marcel Dekker, Inc., 2002, ch. 1.

[141] V. I. Utkin, *Sliding Mode Control: Mathematical Tools, Design and Applications*. Berlin, Germany: Springer, 2008, pp. 289–347.

[142] A. S. Morse, "Supervisory control of families of linear set-point controllers – Part 1: Exact matching," *IEEE Transactions on Automatic Control*, vol. 41, no. 10, pp. 1413–1431, 1996.

[143] ——, "Supervisory control of families of linear set-point controllers – Part 2: Robustness," *IEEE Transactions on Automatic Control*, vol. 42, no. 11, pp. 1500–1515, 1997.

[144] X. D. Koutsoukos, P. J. Antsaklis, J. A. Stiver, and M. D. Lemmon, "Supervisory

control of hybrid systems," *Proceedings of the IEEE*, vol. 88, no. 7, pp. 1026–1049, 2000.

[145] L. Vu and D. Liberzon, "Supervisory control of uncertain linear time-varying systems," *IEEE Transactions on Automatic Control*, vol. 56, no. 1, pp. 27–42, 2011.

[146] C. Carathéodory, *Vorlesungen Über Reelle Funktionen.* Berlin, Germany: Teubner, 1918.

[147] A. F. Filippov, *Differential Equations with Discontinuous Righthand Sides.* Amesterdam, the Netherlands: Springer, 1988.

[148] N. N. Krasovskii, *Stability of Motion. Applications of Lyapunov's Second Method to Differential Systems and Equations With Delay.* Stanford, CA: Stanford University Press, 1963.

[149] F. H. Clarke, Y. S. Ledyaev, R. J. Stern, and P. R. Wolenski, *Nonsmooth Analysis and Control Theory*, ser. Graduate Texts in Mathematics. New York, NY: Springer, 1997.

[150] O. Hájek, "Discontinuous differential equations, I," *Journal of Differential Equations*, vol. 32, no. 2, pp. 149 – 170, 1979.

[151] S. Hu, "Differential equations with discontinuous right-hand sides," *Journal of Mathematical Analysis and Applications*, vol. 154, no. 2, pp. 377 – 390, 1991.

[152] J. S. Spraker and D. C. Biles, "A comparison of the Carathéodory and Filippov solution sets," *Journal of Mathematical Analysis and Applications*, vol. 198, no. 2, pp. 571 – 580, 1996.

[153] J. Cortés, "Discontinuous dynamical systems," *IEEE Control Systems Magazine*, vol. 28, no. 3, pp. 36–73, 2008.

[154] E. M. Stein and R. Shakarchi, *Real Analysis.* Princeton, NJ: Princeton, 2009.

[155] Q. Wang, Y. Hou, and C. Dong, "Model reference robust adaptive control for a class of uncertain switched linear systems," *International Journal of Robust and Nonlinear Control*, vol. 22, no. 9, pp. 1019–1035, 2012.

[156] C. Wu, J. Zhao, and X.-M. Sun, "Adaptive tracking control for uncertain switched systems under asynchronous switching," *International Journal of Robust and Nonlinear Control*, vol. 25, no. 17, pp. 3457–3477, 2015.

[157] J. Xie and J. Zhao, "H_∞ model reference adaptive control for switched systems based on the switched closed-loop reference model," *Nonlinear Analysis: Hybrid Systems*, vol. 27, pp. 92–106, 2018.

[158] Minyue Fu and B. Barmish, "Adaptive stabilization of linear systems via switching control," *IEEE Transactions on Automatic Control*, vol. 31, no. 12, pp. 1097–1103, 1986.

[159] D. E. Miller and E. J. Davison, "An adaptive controller which provides an arbitrarily good transient and steady-state response," *IEEE Transactions on Automatic Control*, vol. 36, no. 1, pp. 68–81, 1991.

[160] K. S. Narendra and J. Balakrishnan, "Adaptive control using multiple models," *IEEE Transactions on Automatic Control*, vol. 42, no. 2, pp. 171–187, 1997.

[161] E. B. Kosmatopoulos and P. A. Ioannou, "A switching adaptive controller for feedback linearizable systems," *IEEE Transactions on Automatic Control*, vol. 44, no. 4, pp. 742–750, 1999.

[162] J. P. Hespanha, D. Liberzon, and A. S. Morse, "Overcoming the limitations of adaptive control by means of logic-based switching," *Systems & Control Letters*, vol. 49, no. 1, pp. 49–65, 2003.

[163] J. Xie and J. Zhao, "Model reference adaptive control for switched LPV systems and its application," *IET Control Theory Applications*, vol. 10, no. 17, pp. 2204–2212, 2016.

[164] J. Xie, S. Li, H. Yan, and D. Yang, "Model reference adaptive control for switched linear systems using switched multiple models control strategy," *Journal of the Franklin Institute*, vol. 356, no. 5, pp. 2645 – 2667, 2019.

[165] C. Tan, G. Tao, R. Qi, and H. Yang, "A direct MRAC based multivariable multiple-model switching control scheme," *Automatica*, vol. 84, pp. 190 – 198, 2017.

[166] B. Paden and S. Sastry, "A calculus for computing Filippov's differential inclusion with application to the variable structure control of robot manipulators," *IEEE Transactions on Circuits and Systems*, vol. 34, no. 1, pp. 73–82, 1987.

[167] N. Fischer, R. Kamalapurkar, and W. E. Dixon, "LaSalle-Yoshizawa corollaries for nonsmooth systems," *IEEE Transactions on Automatic Control*, vol. 58, no. 9, pp. 2333–2338, 2013.

[168] R. Kamalapurkar, J. A. Rosenfeld, A. Parikh, A. R. Teel, and W. E. Dixon, "Invariance-like results for nonautonomous switched systems," *IEEE Transactions on Automatic Control*, vol. 64, no. 2, pp. 614–627, 2019.

[169] J. Xie and J. Zhao, "Model reference adaptive control for nonlinear switched systems under asynchronous switching," *International Journal of Adaptive Control and Signal Processing*, vol. 31, no. 1, pp. 3–22, 2017.

[170] X. Wang, J. Zhao, and Y. Tang, "State tracking model reference adaptive control for switched nonlinear systems with linear uncertain parameters," *Journal of Control Theory and Applications*, vol. 10, no. 3, pp. 354–358, Aug 2012.

[171] Y. Li, S. Sui, and S. Tong, "Adaptive fuzzy control design for stochastic nonlinear

switched systems with arbitrary switchings and unmodeled dynamics," *IEEE Transactions on Cybernetics*, vol. 47, no. 2, pp. 403–414, 2016.

[172] M.-L. Chiang and L.-C. Fu, "Adaptive stabilization of a class of uncertain switched nonlinear systems with backstepping control," *Automatica*, vol. 50, no. 8, pp. 2128–2135, 2014.

[173] S. Vasista, L. Tong, and K. Wong, "Realization of morphing wings: A multidisciplinary challenge," *AIAA Journal of aircraft*, vol. 49, no. 1, pp. 11–28, 2012.

[174] T. M. Seigler, D. A. Neal, J.-S. Bae, and D. J. Inman, "Modeling and flight control of large-scale morphing aircraft," *AIAA Journal of Aircraft*, vol. 44, no. 4, pp. 1077–1087, 2007.

[175] A. Sofla, S. Meguid, K. Tan, and W. Yeo, "Shape morphing of aircraft wing: Status and challenges," *Materials & Design*, vol. 31, no. 3, pp. 1284–1292, 2010.

[176] G. Heredia, A. Duran, and A. Ollero, "Modeling and simulation of the HADA reconfigurable UAV," *Journal of Intelligent & Robotic Systems*, vol. 65, no. 1, pp. 115–122, 2012.

[177] K. Z. Y. Ang, J. Cui, T. Pang, K. Li, K. Wang, Y. Ke, and B. M. Chen, "Development of an unmanned tail-sitter with reconfigurable wings: U-Lion," in *IEEE International Conference on Control Automation*, 2014, pp. 750–755.

[178] H. Cheng, C. Dong, W. Jiang, Q. Wang, and Y. Hou, "Non-fragile switched H_∞ control for morphing aircraft with asynchronous switching," *Chinese Journal of Aeronautics*, vol. 30, no. 3, pp. 1127–1139, 2017.

[179] L. Gong, Q. Wang, and C. Dong, "Disturbance rejection control of morphing aircraft based on switched nonlinear systems," *Nonlinear Dynamics*, vol. 96, no. 2, pp. 975–995, 2019.

[180] X. Jiao and J. Jiang, "Design of adaptive switching control for hypersonic aircraft," *Advances in Mechanical Engineering*, vol. 7, no. 10, p. 1687814015610465, 2015.

[181] L. Marconi, R. Naldi, and L. Gentili, "Modelling and control of a flying robot interacting with the environment," *Automatica*, vol. 47, no. 12, pp. 2571 – 2583, 2011.

[182] L. Marconi and R. Naldi, "Control of aerial robots: Hybrid force and position feedback for a ducted fan," *IEEE Control Systems Magazine*, vol. 32, no. 4, pp. 43–65, 2012.

[183] K. Alexis, G. Darivianakis, M. Burri, and R. Siegwart, "Aerial robotic contact-based inspection: Planning and control," *Autonomous Robots*, vol. 40, no. 4, pp. 631–655, 2016.

[184] D. Liberzon, *Switching in Systems and Control*. Boston, MA: Springer, 2003.

[185] M. Bardi and I. Capuzzo-Dolcetta, *Optimal Control and Viscosity Solutions of Hamilton-Jacobi-Bellman Equations*. Boston, MA: Birkhäuser, 2008.

[186] F. H. Clarke, *Optimization and nonsmooth analysis*. Philadelphia, PA: Society of Industrial and Applied Mathematics, 1989.

[187] H. Federer, *Geometric Measure Theory*. Berlin, Germany: Springer, 2014.

[188] A. S. Morse, "Supervisory control of families of linear set-point controllers-part i. exact matching," *IEEE transactions on Automatic Control*, vol. 41, no. 10, pp. 1413–1431, 1996.

[189] L. Vu, D. Chatterjee, and D. Liberzon, "Iss of switched systems and applications to switching adaptive control," in *IEEE Conference on Decision and Control*, Dec 2005, pp. 120–125.

[190] D. Angeli and D. Liberzon, "A note on uniform global asymptotic stability of nonlinear switched systems in triangular form," in *International Symposium on Mathematical Theory of Networks and System*, 2000.

[191] W. Rudin, *Functional Analysis*. New York, NY: McGraw Hill, 1991.

[192] H. B. Jiang, "Hybrid adaptive and impulsive synchronisation of uncertain complex dynamical networks by the generalised Barbalat's lemma," *IET Control Theory Applications*, vol. 3, no. 10, pp. 1330–1340, 2009.

[193] R. N. Shorten and K. S. Narendra, "On common quadratic Lyapunov functions for pairs of stable LTI systems whose system matrices are in companion form," *IEEE Transactions on Automatic Control*, vol. 48, no. 4, pp. 618–621, 2003.

[194] R. B. Anderson, "Flight experiment of a tilt-rotor quadcopter installing a camera," https://youtu.be/f6wG7IdPXoc, 2019, last accessed 11/29/2019.

[195] F. A. A. (FAA), "Summary of small unmanned aircraft rule (part 107)," 2016.

[196] H. I. Moud, A. Shojaei, I. Flood, X. Zhang, M. Hatami, and M. Rinker, "Qualitative and quantitative risk analysis of unmanned aerial vehicle flights over construction job sites," in *Proceedings of the Eighth International Conference on Advanced Communications and Computation, Barcelona, Spain*, 2018, pp. 22–26.

[197] J. Straube, *Maintenance and Inspection Manual for Precast Concrete Building Enclosures*, RDH building Science, Inc, Ontario, Canada, 2016.

[198] O. Safety, H. Administration *et al.*, "Construction focus four: Fall hazards," *OSHA, US Department of Labor*, 2011.

[199] J. Irizarry, M. Gheisari, and B. N. Walker, "Usability assessment of drone technology as safety inspection tools," *Journal of Information Technology in Construction (ITcon)*, vol. 17, no. 12, pp. 194–212, 2012.

[200] R. R. S. De Melo, D. B. Costa, J. S. Álvares, and J. Irizarry, "Applicability of unmanned aerial system (UAS) for safety inspection on construction sites," *Safety science*, vol. 98, pp. 174–185, 2017.

[201] L. Schreiber and E. Ostiari, "Game of drones: do civilian applications harbour opportunities for sustainable development?" *Mirova. Accessed January 22nd*, 2016.

[202] D. Lattanzi and G. Miller, "Review of robotic infrastructure inspection systems," *Journal of Infrastructure Systems*, vol. 23, no. 3, p. 04017004, 2017.

[203] M. Gheisari and B. Esmaeili, "Unmanned aerial systems (UAS) for construction safety applications," in *Construction Research Congress 2016*, 2016, pp. 2642–2650.

[204] A. Jimenez-Cano, J. Braga, G. Heredia, and A. Ollero, "Aerial manipulator for structure inspection by contact from the underside," in *IEEE/RSJ international conference on intelligent robots & systems*. IEEE, 2015, pp. 1879–1884.

[205] A. Ellenberg, L. Branco, A. Krick, I. Bartoli, and A. Kontsos, "Use of unmanned aerial vehicle for quantitative infrastructure evaluation," *Journal of Infrastructure Systems*, vol. 21, no. 3, p. 04014054, 2014.

[206] R. Ashour, T. Taha, F. Mohamed, E. Hableel, Y. A. Kheil, M. Elsalamouny, M. Kadadha, K. Rangan, J. Dias, L. Seneviratne *et al.*, "Site inspection drone: A solution for inspecting and regulating construction sites," in *IEEE International Midwest Symposium on Circuits and Systems*. IEEE, 2016, pp. 1–4.

[207] M. C. Tatum and J. Liu, "Unmanned aerial vehicles in the construction industry," in *Proceedings of the Unmanned Aircraft System Applications in Construction, Creative Construction Conference, Primosten, Croatia*, 2017, pp. 19–22.

[208] K. Kim, H. Kim, and H. Kim, "Image-based construction hazard avoidance system using augmented reality in wearable device," *Automation in construction*, vol. 83, pp. 390–403, 2017.

[209] M. Gheisari and B. Esmaeili, "Applications and requirements of unmanned aerial systems (UASs) for construction safety," *Safety science*, vol. 118, pp. 230–240, 2019.

[210] J. Howard, V. Murashov, and C. M. Branche, "Unmanned aerial vehicles in construction and worker safety," *American journal of industrial medicine*, vol. 61, no. 1, pp. 3–10, 2018.

[211] A. Plioutsias, N. Karanikas, and M. M. Chatzimihailidou, "Hazard analysis and safety requirements for small drone operations: to what extent do popular drones embed safety?" *Risk Analysis*, vol. 38, no. 3, pp. 562–584, 2018.

[212] K. Afsari, C. Eastman, and D. Shelden, "Building information modeling data interoperability for cloud-based collaboration: Limitations and opportunities," *International Journal of Architectural Computing*, vol. 15, no. 3, pp. 187–202, 2017.

[213] F. M. Arain and M. Burkle, "Learning construction project management in the virtual world: leveraging on second life," *Journal of Information Technology in Construction*, vol. 16, no. 16, pp. 243–258, 2011.

[214] S. Kaewunruen and N. Xu, "Digital twin for sustainability evaluation of railway station buildings," *Frontiers in Built Environment*, vol. 4, p. 77, 2018.

[215] S. H. Lee, S. H. Woo, J. R. Ryu, and S. Y. Choo, "Automated building occupancy authorization using bim and uav-based spatial information: photogrammetric reverse engineering," *Journal of Asian Architecture and Building Engineering*, vol. 18, no. 2, pp. 151–158, 2019.

[216] H. Hamledari, B. McCabe, S. Davari, A. Shahi, E. Rezazadeh Azar, and F. Flager, "Evaluation of computer vision-and 4D BIM-based construction progress tracking on a UAV platform," in *Proc., 6TH CSCE/ASCE/CRC International Construction Specialty Conference*, 2017.

[217] F. Augugliaro, S. Lupashin, M. Hamer, C. Male, M. Hehn, M. W. Mueller, J. S. Willmann, F. Gramazio, M. Kohler, and R. D'Andrea, "The flight assembled architecture

installation: Cooperative construction with flying machines," *IEEE Control Systems Magazine*, vol. 34, no. 4, pp. 46–64, 2014.

[218] C. Papachristos, K. Alexis, and A. Tzes, "Efficient force exertion for aerial robotic manipulation: Exploiting the thrust-vectoring authority of a tri-tiltrotor uav," in *IEEE international conference on robotics and automation*. IEEE, 2014, pp. 4500–4505.

[219] A. Albers, S. Trautmann, T. Howard, T. A. Nguyen, M. Frietsch, and C. Sauter, "Semi-autonomous flying robot for physical interaction with environment," in *2010 IEEE Conference on Robotics, Automation and Mechatronics*. IEEE, 2010, pp. 441–446.

[220] S. Hamaza, I. Georgilas, M. Fernandez, P. Sanchez, T. Richardson, G. Heredia, and A. Ollero, "Sensor installation and retrieval operations using an unmanned aerial manipulator," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2793–2800, 2019.

[221] M. Tognon, H. A. T. Chávez, E. Gasparin, Q. Sablé, D. Bicego, A. Mallet, M. Lany, G. Santi, B. Revaz, J. Cortés *et al.*, "A truly-redundant aerial manipulator system with application to push-and-slide inspection in industrial plants," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1846–1851, 2019.

[222] Q. Lindsey, D. Mellinger, and V. Kumar, "Construction of cubic structures with quadrotor teams," *Proc. Robotics: Science & Systems VII*, 2011.

[223] S.-k. Yun, M. Schwager, and D. Rus, "Coordinating construction of truss structures using distributed equal-mass partitioning," in *Robotics Research*. Springer, 2011, pp. 607–623.

[224] M. Orsag, C. Korpela, S. Bogdan, and P. Oh, "Dexterous aerial robots—mobile manipulation using unmanned aerial systems," *IEEE Transactions on Robotics*, vol. 33, no. 6, pp. 1453–1466, 2017.

[225] P. Chermprayong, K. Zhang, F. Xiao, and M. Kovac, "An integrated delta manipulator for aerial repair: A new aerial robotic system," *IEEE Robotics & Automation Magazine*, vol. 26, no. 1, pp. 54–66, 2019.

[226] M. Tutar and G. Oğuz, "Computational modeling of wind flow around a group of buildings," *International journal of computational fluid dynamics*, vol. 18, no. 8, pp. 651–670, 2004.

[227] H. Ramirez-Rodriguez, V. Parra-Vega, A. Sanchez-Orta, and O. Garcia-Salazar, "Robust backstepping control based on integral sliding modes for tracking of quadrotors," *Journal of Intelligent & Robotic Systems*, vol. 73, no. 1-4, pp. 51–66, 2014.

[228] H. Liu, W. Zhao, Z. Zuo, and Y. Zhong, "Robust control for quadrotors with multiple time-varying uncertainties and delays," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 2, pp. 1303–1312, 2016.

[229] A. L'afflitto, R. B. Anderson, and K. Mohammadi, "An introduction to nonlinear robust control for unmanned quadrotor aircraft: how to design control algorithms for quadrotors using sliding mode control and adaptive control techniques [focus on education]," *IEEE Control Systems Magazine*, vol. 38, no. 3, pp. 102–121, 2018.

[230] B. Zhao, B. Xian, Y. Zhang, and X. Zhang, "Nonlinear robust sliding mode control of a quadrotor unmanned aerial vehicle based on immersion and invariance method," *International Journal of Robust and Nonlinear Control*, vol. 25, no. 18, pp. 3714–3731, 2015.

[231] A. L'Afflitto and K. Mohammadi, "Equations of motion of rotary-wing unmanned aerial system with time-varying inertial properties," *Journal of Guidance, Control, and Dynamics*, vol. 41, no. 2, pp. 559–564, 2018.

[232] B.-C. Min, J.-H. Hong, and E. T. Matson, "Adaptive robust control (ARC) for an altitude control of a quadrotor type uav carrying an unknown payloads," in *2011 11th*

International Conference on Control, Automation and Systems. IEEE, 2011, pp. 1147–1151.

[233] F. A. Goodarzi, D. Lee, and T. Lee, "Geometric control of a quadrotor UAV transporting a payload connected via flexible cable," *International Journal of Control, Automation and Systems*, vol. 13, no. 6, pp. 1486–1498, 2015.

[234] C. Wang, B. Song, P. Huang, and C. Tang, "Trajectory tracking control for quadrotor robot subject to payload variation and wind gust disturbance," *Journal of Intelligent & Robotic Systems*, vol. 83, no. 2, pp. 315–333, 2016.

[235] S. Yang and B. Xian, "Energy-based nonlinear adaptive control design for the quadrotor uav system with a suspended payload," *IEEE Transactions on Industrial Electronics*, vol. 67, no. 3, pp. 2054–2064, 2019.

[236] A. Erasmus and H. Jordaan, "Linear quadratic gaussian control of a quadrotor with an unknown suspended payload," in *2020 International SAUPEC/RobMech/PRASA Conference*. IEEE, 2020, pp. 1–6.

[237] Y. Alothman, W. Jasim, and D. Gu, "Quad-rotor lifting-transporting cable-suspended payloads control," in *International Conference on Automation and Computing*. IEEE, 2015, pp. 1–6.

[238] K. Alexis, G. Darivianakis, M. Burri, and R. Siegwart, "Aerial robotic contact-based inspection: planning and control," *Autonomous Robots*, vol. 40, no. 4, pp. 631–655, 2016.

[239] L. Marconi, R. Naldi, and L. Gentili, "Modelling and control of a flying robot interacting with the environment," *Automatica*, vol. 47, no. 12, pp. 2571–2583, 2011.

[240] Y. Tamura, M. Matsui, L.-C. Pagnini, R. Ishibashi, and A. Yoshida, "Measurement

of wind-induced response of buildings using RTK-GPS," *Journal of Wind Engineering and Industrial Aerodynamics*, vol. 90, no. 12-15, pp. 1783–1793, 2002.

[241] T. Baybura, İ. Tiryakioğlu, M. A. Uğur, H. İ. Solak, and Ş. Şafak, "Examining the accuracy of network rtk and long base rtk methods with repetitive measurements," *Journal of Sensors*, vol. 2019, 2019.

[242] S. Liu, M. Watterson, S. Tang, and V. Kumar, "High speed navigation for quadrotors with limited onboard sensing," in *IEEE international conference on robotics and automation*. IEEE, 2016, pp. 1484–1491.

[243] S. Liu, M. Watterson, K. Mohta, K. Sun, S. Bhattacharya, C. J. Taylor, and V. Kumar, "Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-D complex environments," *IEEE Robotics & Automation Letters*, vol. 2, no. 3, pp. 1688–1695, 2017.

[244] W. Jizhou, L. Zongjian, and L. Chengming, "Reconstruction of buildings from a single UAV image," in *Proc. International Society for Photogrammetry and Remote Sensing Congress*, 2004, pp. 100–103.

[245] F. Yamazaki, T. Matsuda, S. Denda, and W. Liu, "Construction of 3D models of buildings damaged by earthquakes using uav aerial images," in *Tenth Pacific Conference Earthquake Engineering Building an Earthquake-Resilient Pacific; Australian Earthquake Engineering Society: McKinnon, Australia*, 2015, pp. 6–8.

[246] S. Siebert and J. Teizer, "Mobile 3D mapping for surveying earthwork projects using an unmanned aerial vehicle (UAV) system," *Automation in construction*, vol. 41, pp. 1–14, 2014.

[247] Y. Vacanas, K. Themistocleous, A. Agapiou, and D. Hadjimitsis, "The combined use of building information modelling (BIM) and unmanned aerial vehicle (UAV) technologies

for the 3D illustration of the progress of works in infrastructure construction projects," in *Fourth International Conference on Remote Sensing and Geoinformation of the Environment*, vol. 9688. International Society for Optics and Photonics, 2016, p. 96881Z.

[248] P. Glira, N. Pfeifer, and G. Mandlburger, "Rigorous strip adjustment of UAV-based laserscanning data including time-dependent correction of trajectory errors," *Photogrammetric Engineering & Remote Sensing*, vol. 82, no. 12, pp. 945–954, 2016.

[249] Y. K. Cheung, *Finite strip method in structural analysis*. Elsevier, 2013.

[250] C. Eschmann and T. Wundsam, "Web-based georeferenced 3D inspection and monitoring of bridges with unmanned aircraft systems," *Journal of Surveying Engineering*, vol. 143, no. 3, p. 04017003, 2017.

[251] D. Borrmann, A. Nüchter, M. Đakulović, I. Maurović, I. Petrović, D. Osmanković, and J. Velagić, "A mobile robot based system for fully automated thermal 3D mapping," *Advanced Engineering Informatics*, vol. 28, no. 4, pp. 425–440, 2014.

[252] D. González-Aguilera, S. Lagueela, P. Rodríguez-Gonzálvez, and D. Hernández-López, "Image-based thermographic modeling for assessing energy efficiency of buildings façades," *Energy and Buildings*, vol. 65, pp. 29–36, 2013.

[253] N. Haala, M. Cramer, F. Weimer, and M. Trittler, "Performance test on UAV-based photogrammetric data collection," *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 38, no. 6, 2011.

[254] H. Bachmann, A. Steinle *et al.*, *Precast concrete structures*. Wiley Online Library, 2011.

[255] J. Straube, *Maintenance and Inspection Manual for Precast Concrete Building Enclosures*, RDH Building Science, Ontario, Canada, 2016.

[256] A. Das and C. A. Woolsey, "Workspace modeling and path planning for truss structure inspection by unmanned aircraft," *Journal of Aerospace Information Systems*, vol. 16, no. 1, pp. 37–51, 2019.

[257] S. Marschner and P. Shirley, *Fundamentals of computer graphics*. CRC Press, 2015.

[258] S. P. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, UK: Cambridge University Press, 2004.

[259] S. Lin and B. W. Kernighan, "An effective heuristic algorithm for the traveling-salesman problem," *Operations research*, vol. 21, no. 2, pp. 498–516, 1973.

[260] G. Gutin, A. Yeo, and A. Zverovich, "Traveling salesman should not be greedy: domination analysis of greedy-type heuristics for the TSP," *Discrete Applied Mathematics*, vol. 117, no. 1-3, pp. 81–86, 2002.

[261] R. B. Anderson, J. A. Marshall, A. L'Afflitto, and J. Dotterweich, "Model reference adaptive control of switched dynamical systems with applications to aerial robotics," *Journal of Intelligent & Robotic Systems*, 2020.

[262] T. E. Seghier, Y. W. Lim, M. H. Ahmad, and W. O. Samuel, "Building envelope thermal performance assessment using visual programming and BIM, based on ETTV requirement of green mark and greenre," *International Journal of Built Environment and Sustainability*, vol. 4, no. 3, 2017.

[263] P. J. Moriarty and A. C. Hansen, "Aerodyn theory manual," National Renewable Energy Lab., Golden, CO (US), Tech. Rep., 2005.

[264] J. Xie and J. Zhao, "H_∞ model reference adaptive control for switched systems based on the switched closed-loop reference model," *Nonlinear Analysis: Hybrid Systems*, vol. 27, pp. 92–106, 2018.