Monte-Carlo Simulation-based Quadrotor Flight Envelope Prediction

Using Trajectory Pruning with Artificial Neural Networks

Kei Kaneko

Delft University of Technology

TU Delft

Monte-Carlo Simulation-based Quadrotor Flight Envelope Prediction

Using Trajectory Pruning with Artificial Neural Networks

Submitted by

Kei Kaneko

in partial fulfilment of the requirements for obtaining the degree of Master of Science in Aerospace Engineering

Supervisors:	Dr. ir. C. C. de Visser, P. Solanki
Institution:	Delft University of Technology
Place:	Faculty of Aerospace Engineering, Delft

Cover Image Source: `https://unsplash.com/photos/lsqLJCvr0pc`

TU Delft

Preface

This report contains the results of work that was performed since October 2021 as a part of the MSc thesis assignment at the faculty of Aerospace Engineering. The research into a novel approach for flight envelope estimation in an effort to enhance the safety of quadrotor drones has been a challenging yet meaningful experience for me.

Over the duration of the assignment, just under a whole year, I have received a lot of help and support from many different people. I would like to thank my supervisors Coen de Visser and Prashant Solanki for their guidance throughout the period and offering me a chance to participate in this project. I would like to thank all the members of the VIDI group, especially Prashant Solanki for his Level set toolbox and Jasper van Beers for his quadrotor drone simulation environment.

Kei Kaneko
Delft, September 2022

Summary

Quadrotor drones are vehicles with wide range of applications which are gaining more attention in recent years. Although there is a large body of literature on the topic of modelling and controlling of quadrotor drones, prediction of their flight envelopes remains a relatively unexplored research field. The knowledge of the flight envelopes can enhance the understanding of the causes of loss of controls and at the same time aid to prevent them, which could be seen from the application examples on commercial aircraft. Flight envelope prediction is a challenging task where one of the difficulties is that major methods, like the level set methods, is impractical with vehicle models with higher dimensions which is essential to capture the dynamics of highly manoeuvrable quadrotor drones. One method that can be applied to high-dimensional systems is with an approach using Monte-Carlo simulation, which is an algorithm used to estimate probability distributions of a system by calculating outputs with randomly sampled inputs. This can be applied for flight envelope prediction by repeatedly simulating the dynamic system model with randomly sampled control sequences to estimate the reachable set. One of the drawbacks is that a large number of simulations is needed to predict a flight envelope, while not all simulations directly contribute towards estimating its boundary.

A novel approach is proposed in this research that is able to predict the level of contribution towards estimating the reachable set boundary of a given input sample using artificial neural networks, before performing computationally expensive numerical simulations. By rejecting input samples predicted to have relatively low values, the boundary of the reachable set can be reconstructed with a smaller number of simulations. This approach is applied on a longitudinal model of a quadrotor drone in hover with actuator dynamics, which makes it a six-dimensional system. First the reference reachable sets is estimated using Monte-Carlo simulation, which each sample point is labelled a score describing its relative distances from the boundary and from the initial state. A neural network containing long short-term memory cells are trained to map the sampled control sequences to the labelled scores to the end state of corresponding trajectories.

With an appropriate choice of the threshold the trained neural network is able to reject about 50 % of randomly sampled control sequences, in which at most 95% of the rejected samples would not have contributed towards the boundary estimation. By changing the threshold further reduction in the numerical simulation can be achieved with an increased risk of rejecting potentially valuable trajectories. It is further shown that the trained ANN could be applied for trajectory pruning for off-nominal systems with changes to the model parameters. Further research on the computation cost comparison against purely probabilistic approaches is recommended to draw conclusions on the effectiveness of the proposed approach.

Contents

Nomenclature

Abbreviations

ANN Artificial neural network

DFOG Distance field over grid

DT Decision tree

ECEM Extreme control effectiveness method

GA Genetic algorithms

GRU Gated recurrent unit

HJ PDE Hamilton–Jacobi partial differential equation

HJB PDE Hamilton–Jacobi–Bellman partial differential equation

HJI PDE Hamilton–Jacobi–Isaacs partial differential equation

LOC-I Loss of control in flight

LSTM Long short-term memory

MC Monte-Carlo

MISE Mean integrated square error

ODE Ordinary differential equation

RL Reinforcement learning

RNN Recurrent neural network

SL Supervised learning

SVM Support vector machine

UAV Unmanned aerial vehicles

Greek Symbols

α_n Learning rate of the gradient descent in the nth iteration

$\boldsymbol{\delta}$ Control sequence

$\boldsymbol{\theta}_n$ Parameters of an artificial neural network in the nth iteration

$\boldsymbol{\xi}$ State trajectory vector

γ Discount rate of the total reward

μ Membership function

$\Omega_1, \Omega_2, \Omega_3, \Omega_4$	Rotor speeds	rad/s
ϕ	Time to reach function	
π	Policy	
ρ	Reward function	
σ	Activation function	
τ	Time constant of actuator dynamics model	s
θ	Pitch angle	rad

Roman Symbols

$\boldsymbol{d}$	Disturbance vector	
$\boldsymbol{f}$	System dynamics	
$\boldsymbol{u}$	Input vector	
$\boldsymbol{x}$	State vector	
$\boldsymbol{y}_i$	ith sample of random vector of states	
C	Coefficients of aerodynamic forces and moments	
c	Centre coordinate of a zonotope	
E_K	Kinetic energy	J
E_P	Potential energy	J
f_X	Probability density function of the state at $t = T$	
F_x, F_z	Resultant forces in the body frame	N
G	Generator matrix of a zonotope	
g	Gravitational acceleration	m/s^2
H	Hamiltonian function	
h_j	Bandwidth of the jth variable	
I_{yy}	Moment of inertia around the y axis	kg/m^2
J	Objective function	
K	Trim set	
k	Kernel function	
m	Control vector dimension	
m_v	Drone vehicle mass	kg
M_y	Pitching moment in the body frame	N $\cdot$ m

N	Number of samples	
n	State vector dimension	
p_c	Probability of generating a trajectory of length N with constant control inputs	
p_s	Probability of changing the control input in the next simulation step	
q	Pitch rate	rad/s
R	Total reward	
T	Time horizon of the reachable set	s
V	Implicit function representing the reachable set	
V_x, V_z	Velocity in the body frame	m/s
X	Random vector of states	

Superscripts

b	Backward
c	Complement set
f	Forward
$*$	Optimal

Other Symbols

$\dot{\boldsymbol{x}}$	State derivative vector with respect to time
$\hat{H}$	Numerical Hamiltonian
$\mathcal{I}$	Invariant set
$\mathcal{R}$	Reachable set
∇	Vector differential operator
$\tilde{E}$	Probabilistic safe flight envelope

List of Figures

List of Tables

1
Introduction

Multirotor drones are seeing much development in light of their wide applications from reconnaissance to package deliveries. While they become more capable, improving their safety is of high importance. There has been much research on various techniques that makes drones safer to operate. An example is fault tolerant control strategies such as incremental nonlinear dynamic inversion, which also has been applied to drones with different failure modes [1, 2]. Such control strategies require minimal knowledge of current vehicle dynamics to stabilise and control the aircraft. However, further improvement on the safety of aircraft operation can be expected when the knowledge of aircraft's health and safety is available. One measure that serves this purpose is the flight envelope.

A flight envelopes is defined as the region of state space of an aircraft in which the aircraft can operate safely [3]. Meaning that as long as the aircraft states are within these bounds the aircraft has no risk of entering loss of control in flight (LOC-I), which is the dominant cause of incidents for the past decades [4, 5]. Not only can a flight envelope be used as a reliable, quantitative indication of LOC-I [6], more sophisticated applications of flight envelopes are available in the form of flight envelope protection. For instance, a flight controller can limit the control inputs to avoid violations of the flight envelope. These techniques are developed for fixed-wing aircraft [7, 8, 9] as well as for multirotor drones [10]. Flight envelopes are typically determined off-line, prior to operations which there has been research on different methods of estimating the flight envelope of fixed-wing aircraft. Straightforward methods include wind tunnel testing, physical flight tests and high-fidelity model-based computations [11]. More sophisticated methods include formulating the flight envelope prediction as a reachability analysis problem [12], estimating the stability margin through an analysis in the frequency domain [13] and a probabilistic approach with Monte-Carlo simulation [14] to name a few.

Although there has been extensive research performed on the flight envelope prediction methods for commercial aircraft and other forms of fixed-wing aircraft, relatively small body of literature can be found on the topic of flight envelope prediction of multirotor drones. The knowledge of the flight envelope is expected to be of especially high importance for multirotor drones, which is more susceptible to LOC-I due to their nature of being inherently unstable [15]. Furthermore the most popular configuration among multirotor drones, quadrotors, are underactuated systems which can cause challenges in stabilising and controlling the aircraft.

The motivation of this research is to contribute towards improved flight envelope prediction of quadrotor drones, which ultimately could enhance the safety of their operations. The structure of the report is as follows. First, the objective of this thesis is described in Chapter 2, where detailed research questions are derived and motivated. Part I presents the preliminary study performed to gain an understanding of the flight envelope prediction and to identify knowledge gaps. Part II presents the draft scientific paper which contains the main results and conclusions of the research. Part III concludes the thesis as a whole, together with a list of future recommendations. Finally, a book of appendices is presented in Part IV, where the implementations of experiments are described and additional results are presented.

2

Thesis objectives

This chapter describes the objective of this thesis derived from the motivations introduced in Chapter 1 and summarises the findings from the literature study phase. A wide range of flight envelope prediction methods were considered, in which the Monte-Carlo simulation (MCS) based reachability analysis approach was studied in further detail. The goal of this research phase was to gain insight on the overview of the field of flight envelope prediction and to find knowledge gaps in a relevant sub-field which a topic for this research project could be based on. The relevance of the research topic and the feasibility of the research as a MSc thesis project were also considered.

In Section 2.1 the main results from the literature study are summarised. In Section 2.2 the objective of the research and the research questions established through the literature study are presented. Finally in Section 2.3 an overview of the proposed research is described.

2.1. Summary of the literature study

The literature study performed can be divided into four phases. In the first phase, a high-level research was performed in order to grasp the overview of the field of flight envelope prediction and protection techniques. Through this work, it was possible to identify and classify prediction methods into different groups and subgroups, as presented in the classification diagram shown in Figure 3.1. The most popular method among them was identified as the level set method, which its working principles were researched.

In the second phase, several methods that were gaining attention in literature were selected to be studied in more detail in search for knowledge gaps. These were zonotopic reachability analysis, system decomposition techniques, Hamilton-Jacobi PDE solution estimation with neural networks and the MCS approach. From these topics the knowledge gap found in the MCS approach, which is a prediction method involving repeatedly performing numerical simulations with randomly sampled control sequences, were selected to be further investigated. The knowledge gaps found in these topics are briefly discussed in Chapter 3.

The third phase was a more in-depth research on the state-of-the-art of the MCS-based approach which the findings are reported in Chapter 4. From this search it was found that one of the weaknesses of the state-of-the-art by Yin et al. [14] and Sun and de Visser [15] lies in the sampling process for the control sequences, which could potentially be performed in a more efficient way. The last phase of the research was performed in search for enabling technologies that allow more efficient sampling of control sequences for MCS-based reachability analysis. For this purpose, some machine learning techniques were studied in which supervised learning algorithms using artificial neural networks were researched in detail, presented in Chapter 5.

2.2. Research objective and research question

Flight envelope prediction is of high importance for the safety of operations of aircraft but at the same time a challenging task. However, quadrotor drones which are seeing much development still has a relatively small body of literature on their flight envelope predictions. One of them is the work by Sun and de Visser [15] who applied the probabilistic reachability analysis technique using the MCS approach on a quadrotor dynamic model, which a novel sampling strategy of control sequences was demonstrated. However it was observed that there is a potential to be further extended upon, therefore the proposed research objective was established as the following.

Research objective

> *"To further improve the speed and reliability of the Monte-Carlo simulation method for reachability analysis in order to predict the flight envelope of a quadrotor drone by means of effective sampling of control sequences."*

Through literature studies, the capabilities of machine learning to perform complex tasks were deemed to be an enabling technique to realise this research objective. The main research question of this research project was therefore determined as the following.

Main research question

> *"Is it possible to reduce the number of required sample trajectories to predict flight envelopes using the Monte-Carlo simulation approach by integrating machine learning techniques?"*

This research question can be decomposed into different subquestions, in which answering them will give an answer to the main research question. These subquestions are presented below in the intended order to be answered during the research project.

Subquestion 1

> *"What could serve as a reference for the boundary of the theoretical reachable set for a given dynamic system which there is no analytical solution for?"*

Subquestion 2

> *"What measures could be used to distinguish trajectories that reach near the boundaries of reachable sets from those that do not?"*

Subquestion 3

> *"What sort of machine learning technique could be implemented to quickly identify input sequences that contribute more for estimating a more accurate reachable set?"*

Subquestion 4

> *"How does the overall computational complexity compare with other flight envelope prediction methods?"*

2.3. Proposed research

The general framework of the aimed contribution of this research is illustrated in Figure 2.1. The proposed extension to the sampling strategies for flight envelope estimation with the MCS approach is to train a machine that can classify randomly sampled control sequences according to how valuable they are towards constructing the reachable set boundary, which is illustrated on the right side of Figure 2.1. Each relevant field of knowledge to train such a machine are illustrated in the boxes with rounded corners on the left side of Figure 2.1.

In order to quantitatively describe the value of a randomly sampled trajectory, this shall be measured by how close the end state has reached towards an a-priori reachable set. However this requires an knowledge of the true reachable set, which shall be found using the answer to Subquestion 1. Prominent candidates are set constructions with a large number of MCS sample trajectories, either as an α-cut of the fuzzy set found using kernel density estimation as presented by Yin et al. [14] or as an outer surface in three-dimensional space as presented by Sun and de Visser [15]. The next challenge lies in quantitatively describing the value of a given trajectory compared to the identified reference set, as posed in Subquestion 2. Notable candidates are the membership degree of the end state of the trajectory in the fuzzy set representation of the reachable set or the Euclidean norm of the end state from an outer surface constructed by a large number of MC samples. Finally a machine shall be trained that can map a given control sequence to the value of the corresponding trajectory, which is addressed in Subquestion 3 which a prominent candidate is an advanced recurrent neural network architecture such as with long short-term memory cells.

The answers to Subquestions 1, 2 and 3 together enable the aimed contribution shown in Figure 2.1, while the answer to Subquestion 4 can describe how much this research contributes towards the research objective. Identifying valuable input sequences with machine learning techniques still requires computation time as well as training time. It is important to critically study the achieved reduction of computational complexity if any to judge how useful the established method is for flight envelope prediction of quadrotor drones or for other applications.

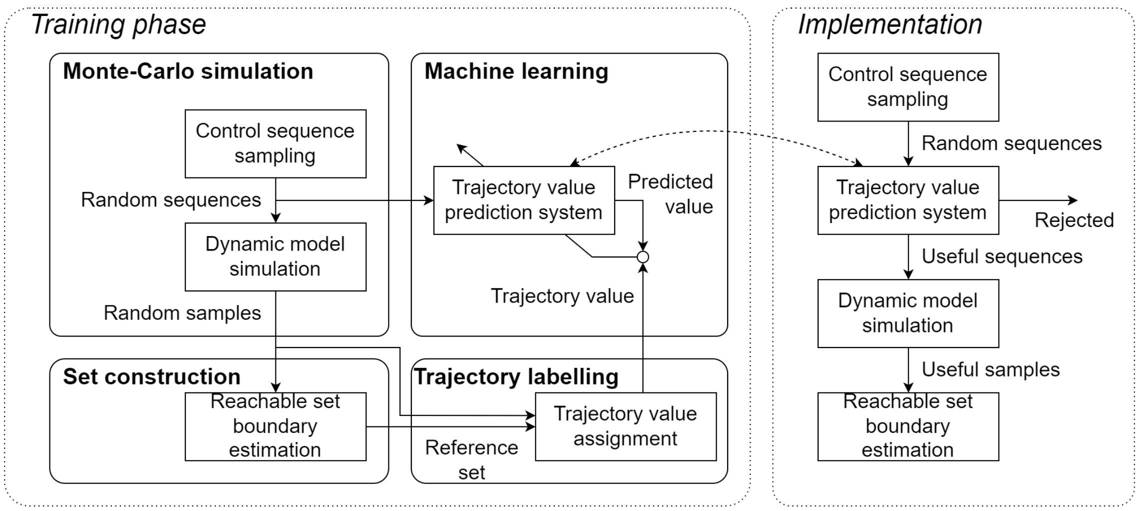


Figure 2.1: An illustration of the aimed contribution of this research. A machine will be trained to filter out less valuable control sequences for reachable set estimation before model simulation (left side), which may be able to save computation power by performing less simulations for MC reachability analysis (right side).

Part I

Preliminary thesis report

3

A review on flight envelope applications and prediction methods

This section presents findings from literature study on different ways flight envelopes can be used to make operations of aircraft safer and different methods of predicting them. In Section 3.1 the definition of the flight envelope used throughout this report is presented. In Section 3.2 different types of flight envelope prediction techniques are introduced. In Section 3.3 the popular approach of representing the flight envelope using reachable sets is introduced. In Section 3.4 the working principle of one of the most studied methods called the level set method is explained. In Section 3.5 some of the knowledge gaps in different approaches of flight envelope prediction identified during the literature study are reported.

3.1. Definition and applications

The term flight envelope is often used to refer to the safe area in the space between aircraft velocity and load factor also known as the V-n diagram. The V-n diagram indicates the boundaries of safe flight conditions with manoeuvrability and structural safety under considerations. However the flight envelope in question serves as a metric with broader application. Although the term does not have a strict definition, the definition by van Oort [3] is adapted throughout this research, which is stated as “the part of the state space for which safe operation of the aircraft and its cargo can be guaranteed and external constraints will not be violated” which is an intersection of following three envelopes: dynamic, structural/comfort and environmental envelopes. The structural/comfort and environmental envelopes pose external constraints, either from the airframe, pilot, passengers and cargo or from the environment such as terrain, walls and other aircraft. These constraints are generally well-known and can be easily quantified [3]. In this report Definition 3.1.1, is adapted and referred as the flight envelope. This definition is also referred as “safe flight envelope” or “manoeuvring envelope” in some literature.

Definition 3.1.1 (Flight envelope/Dynamic envelope)**.** The region of the aircraft state space in which the aircraft can be safely controlled where no loss-of-control events can occur, subject to constraints posed by the dynamic behaviour of the aircraft [3]

For an aircraft represented as a dynamic system in n dimensional state space, the flight envelope can be regarded as an n dimensional volume while its boundary is a $n-1$ dimensional interface. A given point in the state space can be either inside, outside or exactly on the boundary. Flight envelopes can also change their shapes over time, for example due to external disturbances [16] and failure of the aircraft [17].

One primary use of the knowledge of the flight envelope is to gain a better insight in for example a fatal accident, where the flight envelope can be used to make a clear distinction between the LOC-I with other factors of accidents such as Controlled Flight into Terrain [18]. The knowledge of the flight envelope can also be used to describe how severe LOC-I was and to identify possible causes. For instance Wilborn and Foster defined flight envelopes between five different combinations of aircraft state variables, each of them corresponding to five different modes of unsafe flight conditions such as adverse aerodynamics and unusual attitude [6]. They enhance the understanding of how LOC-I occurs in real flights and can serve as foundations of strategies to avoid LOC-I.

The knowledge of the flight envelope is essential for flight envelope protection systems to actively prevent LOC-I. Their functionality is to help pilots to keep the aircraft within the flight envelope [19], by reducing their workload or by enhancing their situation awareness. This can be implemented as human-machine interfaces such as stick shakers and stick pushers [19] or directly to flight control systems to prohibit specific control actions [14]. Given accurate knowledge of the flight envelope and a flight protection system, the number of handling and control accidents in the commercial aviation sector has been greatly reduced [7]. Flight envelopes of quadrotor drones may be applied in a similar fashion to improve the safety of their operations.

3.2. Different flight envelope prediction methods

There exists various methods to predict flight envelopes, which can be categorised into different types and branches as illustrated in Figure 3.1. Methods can mainly be classified into two categories which are test-based methods and model-based methods. Test-based methods include for example wind tunnel testing and physical flight tests, in which the states where an aircraft (or a scale model) can operate safely can be recorded to estimate the flight envelope. Such tests are often time consuming and expensive but are able to produce reliable results. However there may be parts of the flight envelope that are difficult to reach in the given test environment, resulting in a conservative prediction. On the other hand, model-based methods can be performed without the need for physical aircraft, pilot nor special facilities such as the wind tunnel. Model-based methods can perform more extensive analysis at different flight conditions that may be difficult to reach in a real flight. Perhaps the largest drawback of model-based method is that the model almost always contains model gaps as discussed in Subsection 6.2, which creates discrepancies in the actual flight envelope and a perfectly estimated flight envelope of its model. However the estimated envelope could be used to perform physical flights to both validate the envelope as well as to re-identify the model creating an iterative loop to gain further insight in the vehicle dynamics.

Within the model-based approaches, there are again different categories of flight envelope prediction methods. Lichter et al. evaluated the stability margin of an identified dynamic model in frequency domain [13], Shin and Belcasto performed robust analysis to determine a reliable flight regime [20] and Pandita et al. defined the flight envelope as region of attraction [21]. A more rigorous method is to formulate flight envelope prediction as a reachability problem in the optimal control framework, which involves solving Hamilton–Jacobi partial differential equations (HJ PDE) as first demonstrated by Lygeros [12].

The reachability analysis approach is one of the most studied methods for flight envelope prediction with the most popular method called the level set method [22]. The level set method describes the boundaries in a multi-dimensional space using the implicit function, which maps a given state to a numerical value, where the boundaries are all state spaces which the implicit function returns some constant value (often set as zero). The implicit function is typically found with an Eulerian numerical scheme performed over a fixed grid. The level set method has different variations, for example a stochastic approach was taken by van den Brandt et al. [16], semi-Lagrangian approach was taken by van Oort et al. [3] and solved with time scale separation by Kitsios and Lygeros [23]. In recent years, computation methods for HJ PDEs that avoids the computations on fixed grids are attracting interests [24, 25].

Other flight envelope methods in the optimal control framework include for example distance field over grid (DFOG) approach by Helsen et al. [19], zonotopic reachability by Eyang et al. [26] and by solving a time to reach optimisation problem using Fast Marching method by Sethian [27]. Another branch exists in the reachability analysis technique, which takes a probabilistic approach using Monte-Carlo simulation often used for validation purposes. In this approach instead of solving for optimal controls to guide the system from the initial set to another point in state space, the reachable set is estimated by performing a series of numerical simulations using randomly sampled controls. Yin et al. [14] used kernel density estimation to formulate a continuous representation of the reachable set from discrete collection of numerical simulation results who also made use of the extreme control effectiveness method (ECEM) to reduce the sample space of control inputs, while Sun and de Visser [15] extended this method to achieve more effective control sampling strategy.

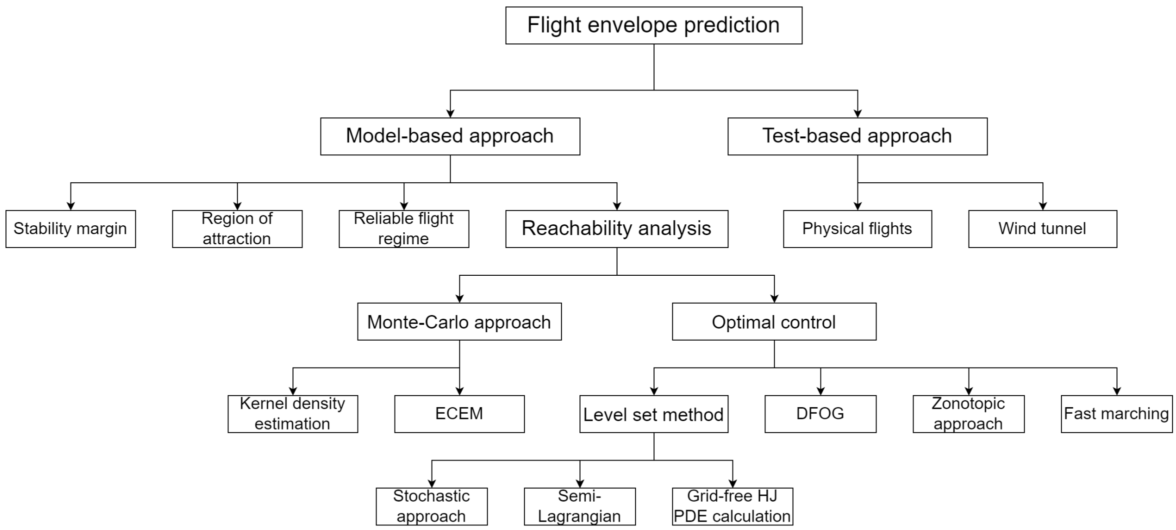


Figure 3.1: A classification diagram of different approaches for flight envelope prediction.

3.3. Flight envelope prediction in terms of reachable sets

One of the most extensively studied methods for flight envelope prediction is by formulating the problem into a reachability problem. Consider a nonlinear dynamic system which its dynamics can be represented as $\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u})$. The mathematical definition of a reachable set by Lygeros [12] is as shown in Equation 3.1. Put into words, a reachable set is the set of states in which there exists an input sequence $\boldsymbol{\delta}$ such that it drives the system into a trajectory ξ which reaches the target set K within the time horizon T. This particular reachable set is referred as the backward reachable set $\mathcal{R}^b$ because for a given trim set K, this set represents the states where the system can return to the trim set within some time horizon T. The definition of a trim set or equivalently a safe set is presented in Definition 3.3.1. Alternatively, this reachable set can be solved backwards in time with the same system with time running backwards as $\dot{\boldsymbol{x}} = -f(\boldsymbol{x}, \boldsymbol{u})$, assuming a continuous dynamic system where there exists a unique trajectory for given initial state, time, and input signal [28]. Then this is referred as the forward reachable set $\mathcal{R}^f$, which are the states the system can reach from a trim set K within some time T. The reachable set is related to another type of set called the invariant set, which is shown in Equation 3.2. For a certain set K, an invariant set is a region in state space where for all possible input sequences the system stays within K. These two sets are related as shown in Equation 3.3 from the duality principle where $(\cdot)^c$ denotes the complement set, meaning that $\mathcal{R}(T,K)$ and $\mathcal{I}(T,K^c)$ share the same boundary.

$$\mathcal{R}(T,K) = \{\boldsymbol{x} \in \mathbb{R}^n \mid \exists \boldsymbol{\delta}(t) \in \mathcal{U}, \exists \tau \in [0,T], \boldsymbol{\xi}(\tau; t, \boldsymbol{x}, \boldsymbol{\delta}) \in K\} \tag{3.1}$$

$$\mathcal{I}(T,K) = \{\boldsymbol{x} \in \mathbb{R}^n \mid \forall \boldsymbol{\delta}(t) \in \mathcal{U}(t), \forall \tau \in [0,T], \boldsymbol{\xi}(\tau; t, \boldsymbol{x}, \boldsymbol{\delta}) \in K\} \tag{3.2}$$

$$\mathcal{R}(T,K) = (\mathcal{I}(T,K^c))^c \tag{3.3}$$

A flight envelope can be predicted by intersecting the forward and the backward reachable sets as shown in Figure 3.2 [3]. This set represents the set of states which an aircraft can travel to and back, starting from an a-priori known trim set K within the time horizon of $2T$. By setting an appropriate time horizon, for example related to the reaction time of a pilot or flight control systems, LOC-I can be effectively avoided by making sure to stay within this intersection. This is a quantitative formulation and satisfies the Definition 3.1.1 presented in Subsection 3.1.

Definition 3.3.1 (Trim set/safe set)**.** The region in the state space which the aircraft is able to maintain its state indefinitely.

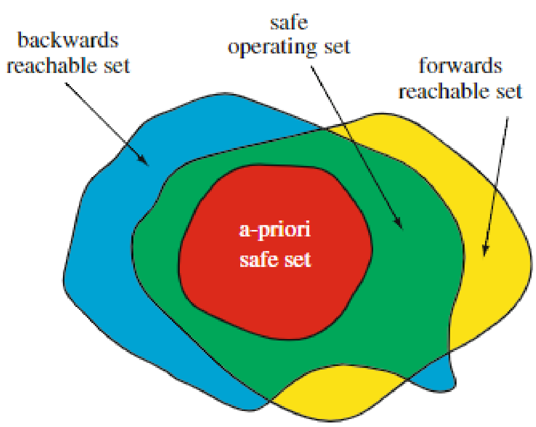


Figure 3.2: Illustration of the flight envelope predicted as the intersection of forward and backward reachable sets [3].

3.4. The level set method

One way of representing a set is using a level set of an implicit function $V : \mathbb{R}^n \to \mathbb{R}$, such that the boundary of a set can be represented as all states $\boldsymbol{x}$, where $V(\boldsymbol{x}) = c$ where c is a constant scalar value, typically taken as zero. Therefore, the reachable set $\mathcal{R}$ of a system can be predicted by solving for an implicit function $V(\boldsymbol{x})$ such that $\mathcal{R} = \{\boldsymbol{x} \in \mathbb{R}^n | V(\boldsymbol{x}) \leq 0\}$. It was shown by Lygeros that a viscosity solution of a HJ PDE can represent the reachable set of a system by formulating the reachability problem as optimal control or differential game theory problem [12]. In a differential game, dynamics depends on the state $\boldsymbol{x}$, the control signal u and the disturbance d in which the goal is to minimise an objective function $J : u, d \to \mathbb{R}$, which is a terminal cost that only depends on the state reached after some time horizon T [12].

A HJ PDE is shown in Equation 3.4, where the second term H is called the Hamiltonian which contains the function that needs to be optimised which depends on the spatial derivative of the value function and the state. The backward and the forward reachable sets are represented as the viscosity solutions of the terminal value and the initial value problems of the HJ PDE, respectively [3]. A wide range of application exists for reachability analysis using HJ PDE, for example in collision avoidance [29], path planning [30] as well as flight envelope prediction of fixed-wing aircraft [3, 31]. For example van Oort et al. solved a type of HJ PDE called the Hamilton–Jacobi–Isaacs (HJI) PDEs shown in Equations 3.5 and 3.6 to solve the backward and forward reachable sets, respectively [3]. Equations 3.5 and 3.6 are terminal value and initial value problems, respectively, where the level sets of $T(\boldsymbol{x})$ and $S(\boldsymbol{x})$ describe the target and the initial sets respectively.

$$\frac{\partial V}{\partial t}(\boldsymbol{x}, t) + H(\frac{\partial V}{\partial x}(\boldsymbol{x}, t), x) = 0 \tag{3.4}$$

$$H^b(\frac{\partial V}{\partial x}(\boldsymbol{x}, t), x) = \min[0, \min_u \max_d \left(\frac{\partial V}{\partial x}\right)^T f(x, u, d)], \quad V(\boldsymbol{x}, 0) = T(\boldsymbol{x}) \tag{3.5}$$

$$H^f(\frac{\partial V}{\partial x}(\boldsymbol{x}, t), x) = \max[0, \max_u \min_d \left(\frac{\partial V}{\partial x}\right)^T f(x, u, d)], \quad V(\boldsymbol{x}, 0) = S(\boldsymbol{x}) \tag{3.6}$$

Such HJ PDEs can be solved using numerical computation schemes developed by Osher and Fedkiw [22], which a MATLAB toolbox developed by Mitchell is often used [32]. The computational complexity of this method is $\mathcal{O}(N^{n+1})$ where n is the number of dimensions and N is the number of nodes in each dimension [33, 34]. The "+1" in the exponent emerges from rounding up the number of time steps N_t as computations are performed on each point on a grid at each time step. These computations include numerically computing the spatial derivatives $\frac{\partial V}{\partial x_i}$ for each dimension i, approximating the numerical Hamiltonian $\hat{H}$, computing the maximum time resolution according to the Courant–Friedrichs–Lewy criteria and numerically solving the resultant differential equation in the form of $\frac{\partial V}{\partial t} + \hat{H} = 0$ [33]. The number of computations required in the level set method increases exponentially with increasing number of dimensions, often referred as "the curse of dimensionality" [35], which limits the number of dimension of a tractable problem to four [10]. This makes the level set method less suitable to be directly applied to flight envelope prediction of highly nonlinear and manoeuvrable systems like quadrotor drones, which lower-dimensional models may not be able to fully capture the dynamics of the system.

The level set method is also known to produce conservative results. Stapel et al. found that a simulation based method predicted 60% faster arrival times compared to the predictions by level set methods when analysing the reachable sets of a simplified aircraft model [34]. This means that control inputs were found through random samples that could reach significantly further than the boundary of the reachable set predicted using the level set method. This is not necessarily harmful, but may overly limit the freedom in manoeuvres performed by pilots and flight control systems. Overly conservative flight envelopes may not be favourable for quadrotor drones, which one of the benefits of the configuration is its high manoeuvrability.

3.5. Knowledge gaps in alternative reachability analysis methods

In the previous sections, the level set method in the optimal control framework of flight envelope estimation technique was described. It was discussed that although the level set method is a well established method with a wide range of application, the high manoeuvrability of quadrotor drones may make the method less suitable as its flight envelope prediction method. In this section, a selection of other reachability analysis techniques that are more suitable for quadrotor configuration that could see further development are mentioned and their knowledge gaps are briefly discussed.

Zonotopic reachability analysis

Sets in space can be represented in different ways in which one of them is by using zonotopes, which is a special case of polytopes defined with two parameters: a centre coordinate c and a generator matrix G. For an n dimensional space, a zonotope is defined as shown in Equation 3.7 [36]. There exists different variations in zonotopes which interested readers are referred to Kochdumper and Althoff's work [37]. Zonotopic representation of a set has a low time and memory complexity thanks to its compact representation [38].

Flight envelope is often defined in many dimensions with many different aircraft states. When predicting flight envelopes with reachability analysis, zonotopes have attractive characteristics to be applied for higher dimensions. Among different types of set representations like ellipsoids and polytopes, computation of reachable sets in terms of zonotopes is relatively more efficient because operations involving zonotopes are closed under a linear transformation and Minkowski summation [39]. Furthermore, it was shown that reachability analysis using a variation called the sparse polynomial zonotopes only involves operation with at most polynomial complexity [37]. One drawback of this method is that it can only be applied to linearised systems, which linearisation errors can act as another uncertainty in the prediction [36].

$$\mathcal{Z} = \left\{ \boldsymbol{x} \in \mathbb{R} | c + \sum_{i=1}^{n} \beta_i G_{(\cdot,i)} \ , \ \beta_i \in [-1,1] \right\} \tag{3.7}$$

Zonotopic reachability analysis for flight envelope prediction has been studied for fixed-wing aircraft in an auto-land mode [26], helicopters [40] and linearised 12 dimensional quadrotor model [39]. This research could be extended to for example quadrotor models with failures. Furthermore, a research on dedicated flight envelope protection systems may follow if flight envelope can be accurately predicted using zonotope representations.

System decomposition to reduce dimensionality of reachability analysis

One way of dealing with the curse of dimensionality is to decompose the dynamic systems with a series of simplifications and assumptions, such that reachability analysis needs to be only performed in lower dimensions. Much like in a way an aircraft can be decoupled in symmetrical and asymmetrical modes, in which dynamic analysis can be performed separately. The method involves two challenges, which are decomposing of dynamic systems and synthesising the reachable sets found in subspaces.

Decomposition techniques have been researched for dynamic models that take specific forms, such as the exact system decomposition method by Chen et al. [41] and the state decoupling disturbance method by Chen et al. [42]. Which the former has been applied to a quadrotor model while an application to quadrotor models for the latter has been mentioned but has not been explicitly presented [42]. The first challenge is that the original dynamic model needs to be converted into respective special forms, which may negatively impact the model fidelity. The second challenge is that after completing reachability analysis separately for each subsystem, intersecting the reachable sets to construct the reachable set of the original system introduces inaccuracies known as the "leaking corners" as Lee et al. pointed out [43].

In order to apply this decomposition technique for higher order systems, an important study is to critically analyse the trade-off between the accuracy of the derived flight envelope and the accomplished reduction of computational complexity. The degradation in the model fidelity through decomposition and making envelopes more conservative to avoid leaking corners may impact the usefulness of the flight envelope for flight envelope protection purposes. Furthermore, application of this technique to quadrotor models with actuator dynamics, which has been shown to have a significant impact on the predicted flight envelope by Sun and de Visser [15], has not been demonstrated to this date.

Estimating viscosity solutions of HJ PDE using neural networks

Level set method has been the most popular method to compute reachable sets to determine a flight envelope of a given aircraft, which involves solving for viscosity solutions of HJ PDE. This requires numerical approximations of high order implicit functions in a grid. As the number of dimensions increases, number of numerical computations needed to cover the grid increases exponentially. One approach to alleviate this is to approximate the viscosity solution in a different way without a need for grid, for example using physics-informed neural networks as a review is summarised by Campbell et al. [44]. Physics-informed neural networks are networks which have been explicitly introduced to the physical laws through symmetries and constraints during training. With such networks, PDEs can be solved using relatively smaller data sets [45], meaning problems involving HJ PDEs may be able to be solved with sparser grids or only with selected samples in the state space.

It has been shown by Darbon et al. that there exist special architectures of neural networks that can represent viscosity solutions of certain classes of HJ PDEs that naturally encode the physics contained in those HJ PDEs [46, 25]. Such methods may be extended to forms of HJ PDE that can be directly applied to reachability analysis, which could be used to flight envelope prediction of higher order dynamic models. However, Campbell et al. cautions that the achieved accuracy with this approach may limit its use to serve only as a reference, which then another prediction method could follow to build upon the estimated envelope from the neural networks [44].

Monte-Carlo simulation

Another branch exists in the reachability analysis methods opposed to the optimal control framework, which is the Monte-Carlo (MC) simulation method. MC simulation is an algorithm used to estimate probability distributions of a system by calculating outputs with sampled inputs based on their distributions, which is commonly used either for validation of analytical models or to solve for complex systems which analytical approximations are difficult to make [47]. This can be applied for flight envelope prediction by repeatedly simulating the system with randomly sampled control sequences. The MC method is a relatively new approach in the field of flight envelope prediction, aside from its uses for validations.

An extensive literature study on this topic was performed, which revealed its high potential to be a suitable flight envelope prediction method for quadrotor drones despite its simplicity. The results of the study are presented in the next chapter in order to identify the research gap present in literature, which this research aims to fill this gap.

4
Flight envelope estimation with the Monte-Carlo approach

As discussed in the previous section, Monte-Carlo (MC) simulation-based reachability analysis is still a relatively new approach applied to flight envelope prediction. In Section 4.1 a general introduction of MC simulation and its application to reachability analysis are presented. In Section 4.2 recent advancements in the MC method shown by Yin et al. [14] and Sun and de Visser [15] are described in detail. In Section 4.3 the state-of-the-art and the knowledge gap present in the field are discussed and summarised.

4.1. General introduction to Monte-Carlo simulation

One approach of estimating the reachable sets is using numerical simulation based techniques, which does not involve solving computationally expensive problems like finding an optimal control to reach a point in space or solving partial differential equations. An example is MC simulation, which is an algorithm used to estimate probability distributions of a system by calculating outputs with randomly sampled inputs, which is commonly used either for validation of analytical models or to solve for complex systems which analytical approximations are difficult to make [47]. MC simulation can be used to estimate reachable sets by repeatedly simulating the dynamic model with randomly sampled control sequence, where each resultant state is a random sample inside the reachable set.

This strategy has been used for verification and validation purposes for the research in the field of reachability analysis and flight envelope prediction [34, 31]. The MC approach remains tractable for higher order systems which level set methods cannot be applied, which is one of the most frequently addressed weaknesses of the level set method. However there are challenges to the MC approach, because in order to make an accurate prediction of a reachable set a very large number of simulations may need to be performed. Furthermore it is also difficult to estimate the required number of simulations to construct the reachable set with a given level of confidence. Another challenge lies in constructing a continuous set from a set of points the simulations were able to reach.

Each numerical simulation performed in the MC approach can be divided into two steps. The first step is to generate a random control sequence of size $m \times N$ by randomly sampling a control action from m dimensional control space at each of the N time steps. The second step is to simulate the system dynamics represented as an initial value problem of a system of ordinary differential equations (ODE) as shown in Equation 4.1 using numerical integration schemes.

$$
\begin{aligned}
\dot{\boldsymbol{x}} &= \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)) \\
\boldsymbol{x}(t_0) &= \boldsymbol{x_0}
\end{aligned}
\tag{4.1}
$$

There exists different types of integration schemes with varying complexities and accuracy depending on for example the number of times the system of ODE $\boldsymbol{f}$ is being evaluated at each time step. One example of numerical methods for solving systems of ODE is the 4th order Runge–Kutta method which evaluates $\boldsymbol{f}$ four times per time step. If higher accuracy is required, discretisation error can be reduced by employing more advanced methods such as higher-order Runge–Kutta methods which comes at a cost of increased number of evaluations of f at each time step.

Furthermore, the setting of the time step for the numerical method for solving ODEs can have an effect on the stability of the solution. When the time step is too large, the approximation error can continuously grow over time, which results in an inaccurate approximation. Suppose that the control sequences are sampled with a relatively large time interval, then a smaller time step than this interval may need to be applied for numerically solving the ODE. Therefore, the number of computations needed to generate one sample point inside the reachable set of a system depends on the system dynamics as well as the required accuracy.

There exists different numerical schemes to solve ODEs which in itself forms a large body of literature. In terms of development of the MC simulation-based reachability analysis, another approach is the seek for more efficient sampling strategies to sample control sequences. In the next section, findings from the literature review on the state-of-the-art of flight envelope prediction method using the MC approach are presented.

4.2. State-of-the-art of Monte-Carlo flight envelope prediction

Although the MC flight envelope prediction suffers less from the curse of dimensionality, which is essential for applications using more complex high dimensional systems, there exist areas of continued development for flight envelope prediction. In recent years, research has been performed to reduce the number of simulations to be performed for reachable set estimation with the MC approach. In this section, the state-of-the-art methods in this field by Yin et al. [14] and Sun and de Visser [15] are presented in Subsections 4.2.1 and 4.2.2 respectively.

4.2.1. Probabilistic flight envelope and extreme control effectiveness method

Yin et al. performed a study on estimation and protection of flight envelope of a seven-dimensional nonlinear simulation model of Lockheed Martin's Innovative Control Effectors aircraft [14]. A novel flight envelope prediction method was applied in their work, which uses kernel density estimation to produce a fuzzy flight envelope assuming the final states of simulated trajectories as stochastic variables.

Flight envelopes are often treated as a crisp set, meaning according to the envelope an aircraft is either "safe" or "not safe" and never in between. Yin et al. argue that this is not a practical definition, as the safety of the aircraft depends on various factors such as external disturbances, pilot's ability to recover the aircraft, modelling errors and controller performance [14]. When the flight envelope is modelled as a fuzzy set the degree of membership of a given state $\boldsymbol{x}$ is described with a membership function $\mu_{\tilde{E}}(\boldsymbol{x})$, which returns a value between 0 (not safe) and 1 (safe). The probabilistic safe flight envelope $\tilde{E}$ can be described using this concept, which is shown in Equation 4.2 defined per time horizon T and target set K. $f_X(\boldsymbol{x})$ is the probability density function of the state at $t = T$ for all safe trajectories of time horizon $2T$.

$$
\tilde{E}(T, K) = \left\{ \left(\boldsymbol{x} \in \mathbb{R}^n, \mu_{\tilde{E}}(\boldsymbol{x}) = \frac{f_X(\boldsymbol{x})}{\max_{\boldsymbol{x} \in \mathcal{R}^n} f_X(\boldsymbol{x})} \right) \right\}
\tag{4.2}
$$

The main challenge is to find f_X, which Yin et al. [14] estimated using the kernel density estimator. Suppose a n dimensional random vector $\boldsymbol{X} = (X_1, X_2, ..., X_n)$ and N samples from it where the ith sample is denoted as $\boldsymbol{y}_i = (y_{i,1}, y_{i,2}, ..., y_{i,n})^T$. Then with these samples, the kernel density estimator is shown as in Equation 4.3 where $K_H(\cdot)$ is the kernel, which is a distribution estimation with 1 sample at the origin. This can be normalised as shown in Equation 4.4, where $k(\cdot)$ is a normalised Gaussian kernel function and h_j is a bandwidth which is equivalent to bin size in a histogram. Therefore this approach constructs a more informative representation of flight envelope as a fuzzy set and at the same time create a continuous set using scattered end state points.

$$\hat{f}(\boldsymbol{x}) = \frac{1}{N}\sum_{i=1}^{N} K_H(\boldsymbol{x} - \boldsymbol{y}_i) \tag{4.3}$$

$$\hat{f}(\boldsymbol{x}) = \frac{1}{N \cdot h_1 \cdot h_2 ... h_n}\sum_{i=1}^{N}\prod_{j=1}^{n} k(\frac{x_j - y_{ij}}{h_j}) \tag{4.4}$$

Another innovation presented was an efficient sampling strategy for the control sequences to simulate trajectories called the extreme control effectiveness method (ECEM) [14]. Yin et al. proved that for input-affine systems an optimal control to reach a certain state within a given time limit is almost always with extreme control effectiveness, making full use of the control surfaces to drive the system in a certain direction. This observation can be made by deriving the Hamilton–Jacobi–Bellman PDE (HJB PDE) , which is a variation of HJI PDE which assumes no disturbances d. Considering an input-affine system ($\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u}) = b(\boldsymbol{x}) + A(\boldsymbol{x})\boldsymbol{u}$), the optimal control of ith control element is as shown in Equation 4.5. This shows that when the control sequence has any effect on the Hamiltonian of the HJB PDE, meaning when $\frac{\partial V}{\partial \boldsymbol{x}}A(\boldsymbol{x}) \neq 0$, the optimal control u_i^* is always either of the extreme ends of the control. With this proof, Yin et al. showed that it is possible to greatly reduce the number of simulations to determine the reachable sets by narrowing down the input space.

$$u_i^* = \begin{cases} u_{i,\max}, & \dfrac{\partial V}{\partial \boldsymbol{x}}A(\boldsymbol{x}) < 0 \\ u_{i,\min}, & \dfrac{\partial V}{\partial \boldsymbol{x}}A(\boldsymbol{x}) > 0 \end{cases} \tag{4.5}$$

Yin et al. also mentions that the required sample size N_{req} to maintain a certain level of mean integrated square error (MISE) of the constructed reachable set relates to dimension n as shown in Equation 4.6 [48]. This suggests that the MC approach also suffers from the curse of dimensionality, but to a lesser extent than numerical methods performed on grids like the level set method. However, although difficult to find analytically, it may be possible to further reduce the scaling by not only randomly sampling the control sequences but to only sample sequences that results in samples inside the reachable set that reaches further towards the boundary.

$$N_{req} = \mathcal{O}(\text{MISE}^{\frac{-(4+n)}{4}}) \tag{4.6}$$

4.2.2. Application on quadrotor drone models

Application of the MC approach for flight envelope prediction of quadrotor drone models has been demonstrated by Sun and de Visser [15]. The model used was a gray-box six-dimensional longitudinal model identified partially using a stepwise regression method [49] with added actuator dynamics, which means that the level set method is intractable for this problem. Sun and de Visser argued that the flight envelope of a highly manoeuvrable system like quadrotor drones should be estimated with a shorter time horizon than for example fixed-wing aircraft, otherwise the flight envelope will be estimated as an arbitrarily large set which does not serve any significant meaning. It was also shown that the modelling of the actuator dynamics has a significant effect on the predicted reachable sets with briefer time horizons as shown in Figure 4.1, which suggest that the inclusion of actuator dynamics in the dynamic model is essential for quadrotor drone flight envelope prediction.

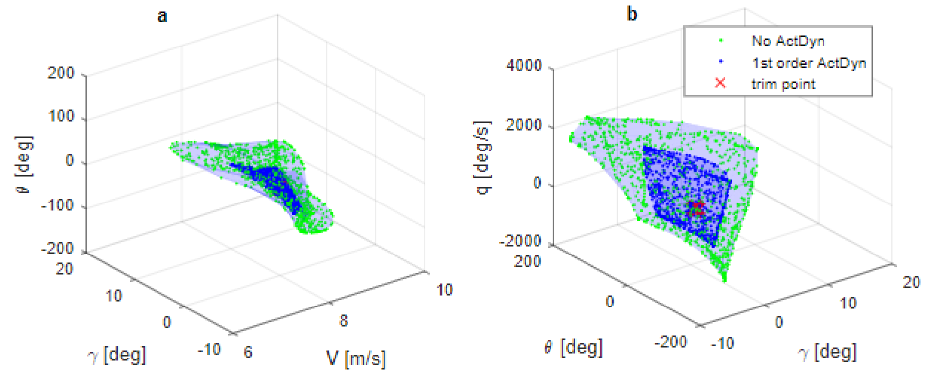


Figure 4.1: Estimations of forward reachable sets of a quadrotor with (blue) and without (green) modelling the actuator dynamics [15]. It can be seen that the reachable sets are smaller for models with actuator dynamics.

Sun and de Visser adapted the ECEM by Yin et al. [14] and achieved further reduction in the required sample trajectory by introducing a parameter that determines the probability of switching between the two extreme input values in each dimension of the control space [15]. This is mathematically shown in Equation 4.7, where p_s is the probability the next control input is sampled as the opposite control to the current control. Consequently, the probability of sampling a constant control input sequence of length N is shown in Equation 4.8. This is a simple yet effective way to increase the chance of reaching the set boundary with each sample of control sequences with an appropriate choice of p_c. For instance by setting p_c as a relatively large value, it is possible to construct input sequences with higher chances of maintaining one extreme control input leading to control sequences that deviate more from its original trim state. This can be seen from Figure 4.2 which shows the end points reached after simulating trajectories sampled with different values of p_c. Larger value of p_c yields relatively more control sequences that result in trajectories which reach further towards the boundary of the reachable set.

$$p_s = 1 - P(u_i(k+h) = u_i(k)) \tag{4.7}$$

$$p_c = (1 - p_s)^N \tag{4.8}$$

4.3. Summary of previous works and knowledge gaps

In the previous sections, the general working principle of reachability analysis with the MC approach and the state-of-the-art application on flight envelope prediction were presented. It was shown that the MC approach can be extended beyond serving only as a validation tool but as a prediction technique that is strictly conservative which can be applied to higher

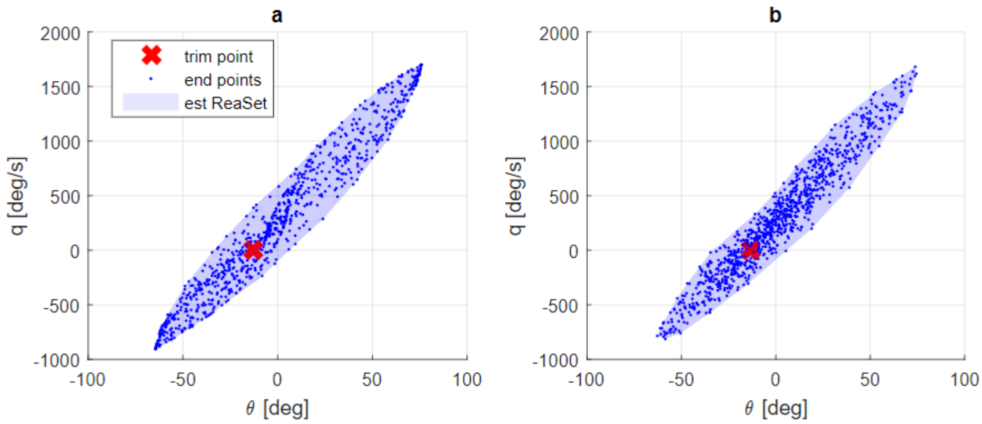


Figure 4.2: Forward reachable set projection of a quadrotor on $q - \theta$ plane sampled with ***a***.) $p_c = 0.1$ and ***b***.) $p_c = 0.001$ [15]

dimensional vehicle models. However there remain unresolved challenges, for example since this approach is the only tractable method for nonlinear problems over certain number of dimensions, it is not possible to validate or compare the results with other methods. This makes it even more difficult to assess the reliability and the level of conservativeness of a given prediction.

One approach of improving the current state-of-the-art is to make better use of the sampled trajectories. Current state-of-the-art only takes the end states of the trajectories into account and not the states reached before the end of the time horizon. For example a trajectory with its end state in the trim set might have explored the flight envelope further away towards the boundary before moving back to the trim set, then this trajectory could have provided more valuable information than the state it reached at the end. This may be applied for more efficient exploration by simulating a series of trajectories that have identical initial sequences, which the numerical simulation of the initial part need to be only performed once.

Campbell et al. conducted a study on existing flight envelope prediction techniques that could be applied to quadrotor drones, in which they raised three shortcomings of the flight envelope prediction method of Yin et al. [44]. First, the framework initiates with an optimisation problem to determine the trim point, which limits the exploration of the trim set and requires significant amount of computing time. Second, the MC sampling approach introduces uncertainty with the required search completion time. Finally, the construction of the flight envelope is done with kernel density estimation, which limits the dimensionality of the estimator and assumes that the forward reachable set has been estimated with high confidence. In terms of taking a probabilistic approach using MC simulations, the second point raised by Campbell et al. is a highly relevant issue. Further improvement on the efficiency of the sampling strategy may alleviate this issue, which Campbell et al. suggest to consider employing machine learning techniques, such as continuous reinforcement learning techniques, which may enable further reduction the required number of simulations to predict reachable sets [44]. Such techniques may enable sampling strategies that only samples trajectories that drives the system further away from the trim set regions that have not been explored by previous samples.

Although MC simulation has been extensively used for validation purposes, less attention was paid on using it as the primary method to compute reachable sets. A large contribution to why MC method has not been used for flight envelope prediction outside of validation is the lack of ways to quantitatively assess the accuracy of the prediction. This made the flight envelopes to be only predicted with "large enough" sample sizes such that the prediction converges “enough” to the true envelope. Yin et al. [14] and Sun and de Visser [15] showed that it is possible to reduce this required number of samples by reducing the sample space and by using an efficient sampling strategy. However this could be further improved by only sampling control sequences that can be used to efficiently construct reachable sets. To summarise, the MC approach is only able to predict a meaningful flight envelope with a large sample size, which this number may be further reduced with the help of machine learning techniques.

5

Machine learning techniques for enhancing the Monte-Carlo approach

Machine learning is a type of artificial intelligence that involves training a machine or a software to produce desired outcomes. There are a variety of tasks that can be performed with machine learning techniques such as classification, regression, optical character and speech recognition, machine translation, denoising and density estimation to name a few [50]. In this section, some of the machine learning techniques that could be used for flight envelope prediction using MC simulation are introduced in which one particular technique is described in further detail.

In Section 5.1 relevant types of tasks that machine learning techniques can be used are mentioned and their relations to flight envelope prediction are discussed. In Section 5.2 different types of machine learning algorithms and relevant models are discussed for their applicability for flight envelope prediction. In Section 5.3 the working principles of artificial neural networks are described in further detail.

5.1. Relevant machine learning tasks for flight envelope prediction

Among different types of machine learning tasks, some are more relevant for flight envelope prediction with the MC approach. In this section, relevant tasks are presented together with how they could be used to enhance the flight envelope prediction process.

Classification

Classification is a task where the machine is asked to classify the input $\boldsymbol{x}$ into a predefined set of categories. A classification function f with k categories with an n dimensional input can be expressed as $f : \mathbb{R}^n \rightarrow \{1, 2, ..., k\}$, while it is also possible to construct a function that return the probability distribution over each category [50]. A common application for this type of task is object recognition, for example identifying whether a given image has either a dog or a cat in it.

This type of task could be used in the process of MC simulations, for example by classifying input sequences that are more likely to stay near the initial state and those that are more likely to drive the system towards the boundary of the reachable set. This classification function may be used as a filter to omit spending computational time on simulating control sequences that do not contribute to estimate the reachable set boundary.

Regression

Regression is a task where the machine is trained to predict a numerical value to a given input with a function $f : \mathbb{R}^n \to \mathbb{R}$ [50]. This is similar to the classification task, but the output format can be any real scalar value. An example of its applications is parameter estimation in system identification which approximates the relationship between the dependent and independent variables.

Regression could for example be used to train an inverse model that returns a corresponding input sequence to a specified target state, which could be used to effectively explore the state space to form reachable sets.

Density estimation

With this task the machine is asked to learn a function $p : \mathbb{R}^n \to \mathbb{R}$ that represents a probability density function [50]. The machine must learn where in the state space some event is likely to occur, which in the case of reachability analysis this could be used to describe where the system can certainly reach ($p(\boldsymbol{x}) = 1$) and where the system certainly cannot reach ($p(\boldsymbol{x}) = 0$) in a given time horizon, like how Yin et al. [14] constructed a fuzzy representation of the reachable set using kernel density estimation.

This could be used to study the effectiveness of a given control sequence sampling strategy. This can for example be done by monitoring the convergence of the estimated probability density representing the reachable set. The probability density estimated using an efficient sampling strategy should statistically converge with smaller number of simulation samples compared to the one estimated using random samples.

5.2. Suitable types of learning

There exists mainly three branches of learning strategies. These are unsupervised learning, supervised learning and reinforcement learning. The main differences between them are the amount of feedback given to the machine in their learning processes. In unsupervised learning the machine does not receive any feedback and learns by itself, in supervised learning the machine has access to the explicit solution it is expected to replicate, while in reinforcement learning the machine receives feedback on their performance in the form of reward. Out of these types of learning strategies, unsupervised learning is not suitable for application as there exists, although not explicitly defined quantitatively, a clear objective in this research which is to reduce the sample trajectories in flight envelope prediction using the MC approach.

In the remainder of this Section, supervised learning and reinforcement learning are described in detail and their potential applications to flight envelope prediction are discussed in Subsections 5.2.1 and 5.2.2. In Subsection 5.2.3 some common machine learning algorithms and models are introduced, in which a suitable algorithm and the model type are selected.

5.2.1. Supervised learning

Supervised learning (SL) is a type of machine learning technique in which a machine is tasked to map a given input to either a specific target output or a label. A SL algorithm analyses the training data and produces a function that can model the relationship between the inputs and the outputs. This function can then be used to approximate the output of the corresponding relation for an unseen input. SL algorithms can be used to perform classification and regression tasks, where the expected label or output are known.

In the case of flight envelope estimation, SL algorithms could be used to approximate the value of a given control sequence according to some function that quantitatively describes its corresponding trajectory's value. For example each end state of a trajectory can be assigned a scalar value v to describe its contribution in constructing the reachable set boundary, in which

SL algorithm can be used to perform regression to map the corresponding control sequence $\boldsymbol{\delta}$ to this scalar value to learn the function $f : \mathbb{R}^{N\times m} \rightarrow \mathbb{R}$ such that $f(\boldsymbol{\delta}) = \hat{v}$. It is also possible to perform classification by setting a label to each control sequence as "valuable" or "not valuable", for example according to the said scalar value. To summarise, SL algorithms could be used to construct the function that directly predicts the value of a given control sequence without performing numerical simulation nor explicitly calculating the value from the reached end state, which an illustration of it is shown in Figure 5.1. Such a machine can be used to filter out control sequences that are unlikely to contribute towards construction of the reachable set construction and to speed up the prediction process by saving computations.

There are two foreseeable challenges in this approach. The first challenge is the establishment of the criteria in judging how valuable a given end state is. When estimating the reachable set, the true reachable set is unknown which makes it extremely difficult to assess whether a given point in the state space is relatively far from the trim set or not. Furthermore, the value of a trajectory can also change depending on which end states have already been sampled. Even if a sampled control sequence drives the system near the boundary of a reachable set, it is of less value when there has already been a sample that drives the system in a neighbourhood.

The next challenge is determining a suitable representation for the input sequence. The most straightforward approach is to set the number of inputs as T/dt where T is the time horizon and dt is the time interval forming a whole sequence of control inputs. However this formulation can become very large depending on the problem. There may be a more efficient, compact representation of control sequences which helps the machine to learn with a higher accuracy and with a shorter training time.

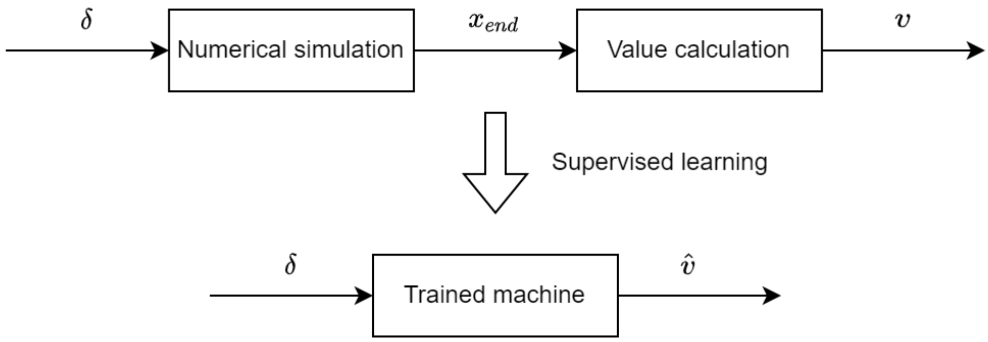


Figure 5.1: A diagram that illustrates how SL algorithms could be used to train machines that directly computes the approximate value $\hat{v}$ of the corresponding trajectory for a given control sequence $\boldsymbol{\delta}$.

5.2.2. Reinforcement learning

Reinforcement learning (RL) is a type of machine learning technique that can be used to train machines to perform various tasks, where an agent learns according to the reward which describes the performance. A trained agent performs some action $\boldsymbol{u}$ depending on the feedback state $\boldsymbol{x}$ from the process. How well the machine performed is assessed by the reward function that considers the state and or the action.

The goal of RL is to find the optimal policy $\pi^*(\boldsymbol{x})$ that maps the optimal action $\boldsymbol{u}^*$ to a given state $\boldsymbol{x}$. Conventional RL method requires the action and the state spaces to be discretised, so that the policy can be described as $\pi(\boldsymbol{x}) : X \rightarrow U$ where X and U are discretised state and action spaces respectively. How well an agent is performing can be measured with the output of the reward function $\rho(\boldsymbol{x}, \boldsymbol{u})$, which can be classified into immediate reward and total reward.

As the name suggests, former only considers the achieved state in the next time step while the latter considers the rewards in the long run. It is possible to express the total reward with the discounted return R^π for the optimal policy as shown in Equation 5.1, where γ is the discount rate which can adjust how much of the future reward is considered at each time step.

$$R^\pi(\boldsymbol{x}_0) = \sum_{k=0}^{\infty} \gamma^k \rho(\boldsymbol{x}_k, u_k), \text{ where } u_k = \pi(\boldsymbol{x}_k) \tag{5.1}$$

In RL optimality is measured based on the expected return which can be defined in two ways, either as a state value function $V^\pi(\boldsymbol{x}_0) = R^\pi(\boldsymbol{x}_0)$ or as a state-action value function $Q^\pi(\boldsymbol{x}_0, \boldsymbol{u}_0) = \rho(\boldsymbol{x}_0, \boldsymbol{u}_0) + \gamma R^\pi(\boldsymbol{x}_0)$. When training an agent with state-action value function, not only the achieved states but the actions are also considered when assessing the performance.

An example in which RL algorithms could be used to improve flight envelope prediction is to train a machine that can find the control inputs at a given state to move further away from the trim set. This was also suggested by suggested by Campbell et al. [44], which could make the sampling process in the work of Yin et al [14] more efficient which employed completely random sampling. Although RL require training time, which may be significantly longer than randomly generating control sequences and simulating them, the trained machine may be used for other quadrotor models. With RL algorithms it may be able to teach the agent an “intuition” in which humans may have to make quadrotor drones perform more extreme manoeuvres. Although this approach goes outside the scope of the MC approach which primarily uses random samples, this may be an effective alternative method to generate control sequences for simulation-based reachability analysis.

5.2.3. Common machine learning algorithms

In the previous Subsections, possibilities of integrating SL or RL algorithms into flight envelope prediction using the MC approach were considered. While RL algorithms may be able to enhance simulation-based reachability analysis it deviates beyond the scope of flight envelope prediction in the MC approach, it was seen that SL algorithms may be used in conjunction with the MC approach. In this section, suitable SL algorithms that could be implemented to fill the identified knowledge gaps are discussed. These are for example support-vector machine (SVM), decision tree (DT) and artificial neural network (ANN) which all can be used to perform both classification and regression tasks.

SVM maps the input vector into a high dimensional feature space, which a linear surface is constructed in this space that separates the data with the largest margin [51]. DT categorises a given input by passing through a series of tests that split inputs depending on their features, which can produce an intuitive classification function but are difficult to be trained [51]. ANN can be used as a nonlinear model which can be trained to approximate the relation between arbitrary sized input and output vectors [50].

Another technique that could potentially be used for in a simulation-based reachability analysis is genetic algorithms (GA), which is a metaheuristic technique inspired by the process of natural selection. The basic elements of GA are points in the solution space represented as chromosomes, fitness selection and biological-inspired operators (crossover, mutation and selection). GA is known to have two main advantages, which are its ability to deal with complex problems and being able to explore search spaces in parallel [52]. These are relevant for simulation-based flight envelope prediction as it most likely involves complex, nonlinear dynamic systems in which a collection of sample trajectories are needed to formulate a reachable set. GA may be used to generate and develop a population of control sequences that can be used to effectively estimate the reachable set.

One disadvantage of GA is that inappropriate choices of population size, fitness function and parameters of biological-inspired operators can lead the algorithm to not converge or to not produce any meaningful result [52]. Tied to the property of simulation-based reachability analysis that there exists no explicit "optimal" trajectories and how reachable sets can only be constructed with multiple samples of trajectories, application of GA to aid the MC sampling may need to be performed mostly in a trial-and-error manner.

Application of supervised learning techniques for classification or regression tasks on sequences of control inputs for simulation-based reachability analysis could not be found in literature at the time of writing. Within these different options, ANN may be a suitable model that has a high approximation power while there also exists specialised network architectures for processing sequential data. The working principles of ANN, training processes and different variations are described in more detail in the next Section.

5.3. Artificial neural network

ANN is a type of model that mimics how a human brain operates with biological neural network, which consists of interconnected neurons sending electrical signals to each other. ANN takes in some input which is passed on to a layer of neurons, which are contain nonlinear functions, which the weighted sums of the neuron outputs are the outputs of the model.

5.3.1. Artificial neuron

An illustration of how each artificial neuron passes on its inputs to the next layer is described in Figure 5.2. Each neuron takes in a weighted sum of the inputs from the previous layer which then passes through the activation function $\sigma(z)$, which is often a nonlinear function. The activation function can take many forms, which can be classified as a projection function when it has a global effect across z or a kernel function which only has local effects [50]. Some examples of activation functions are sigmoid function, rectified linear unit (ReLU) function and exponential linear unit (ELU), which are shown in Equations 5.2 to 5.4 where α is a constant in Equation 5.4. An example of a kernel function is a Gaussian radial basis function, shown in Equation 5.5. Each activation function has unique shapes that may be favourable for certain modelling applications.

$$\sigma(z) = \frac{1}{1+e^{-z}} \tag{5.2}$$

$$\sigma(z) = \begin{cases} 0 \text{ if } z < 0 \\ z \text{ if } z \geq 0 \end{cases} \tag{5.3}$$

$$\sigma(z) = \begin{cases} z \text{ if } z > 0 \\ \alpha(e^z - 1) \text{ if } z \leq 0 \end{cases} \tag{5.4}$$

$$\sigma(z) = e^{-z^2} \tag{5.5}$$

5.3.2. Network of neurons

An artificial neuron does not serve a lot of purpose when considered by itself, however when multiple neurons are inter-connected into a network they can together form a very versatile model. ANN is an extremely powerful function approximating tool which can also be seen from the famous theorem by Cybenko [53] which states that a feedforward ANN with at least one hidden layer with sigmoid activation functions can approximate any continuous nonlinear function arbitrarily well on a compact set, provided that a sufficient number of hidden neurons

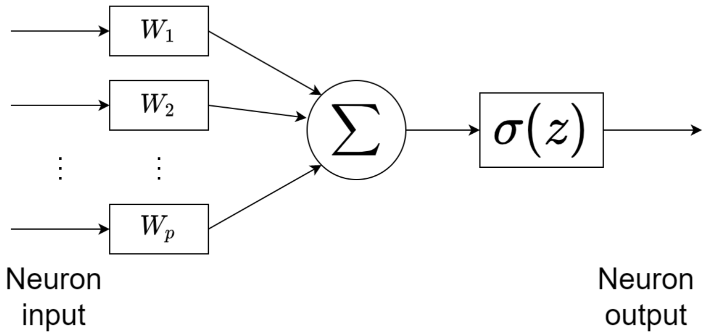


Figure 5.2: An illustration describing the mathematical operation performed at each artificial neuron in an ANN.

are available. A feedforward ANN with one hidden layer is one of the most generic architectures of ANN, yet it has the capability to model any nonlinear functions. Furthermore different types of architectures are available, that may be more suitable to model certain forms of functions. In the remainder of this subsection, different architectures of ANNs that could be used to classify MC samples of control sequences for predicting flight envelopes are presented.

Feedforward neural networks

This is an architecture which an input is fed through the layer of neurons and fed out of the network as an output without being redirected back into the network. An illustration of a feedforward ANN is shown in Figure 5.3 where the input $\boldsymbol{x}$ flows through the network from left to right which at the end gets fed out as the output $\boldsymbol{y}$. The architecture consists of three layers, in which nonlinear activation functions are only used in the neurons inside the hidden layer, meaning that the input layer and the output layer have neurons that simply feed out weighted sums of the previous layer outputs. The main parameters that can be adjusted during the training process is the weights before layers, denoted as W^h for the hidden layer weights and W^o for the output layer weights. This ANN architecture can be modified with the number of neurons inside the hidden layer and the type of activation functions. It is also possible to have multiple hidden layers, which are referred as "deep" feedforward networks.

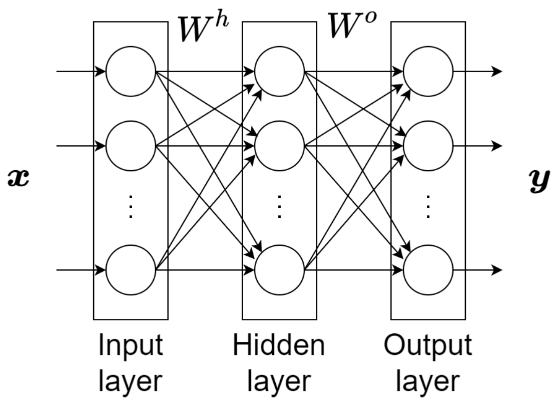


Figure 5.3: Diagram representing a feedforward neural network architecture, which consists of three layers of neurons.

Recurrent neural networks

Recurrent neural network (RNN) is an architecture which is suitable for processing sequential data in the form of $x^{(1)}, x^{(2)}, ...x^{(n)}$. The architecture formulation exploits a network that share same model parameters, where the output of a neuron layer is fed back to itself as shown in Figure 5.4. With this formulation, it is possible to reduce the number of model parameters, namely the weights, to achieve highly compact formulations.

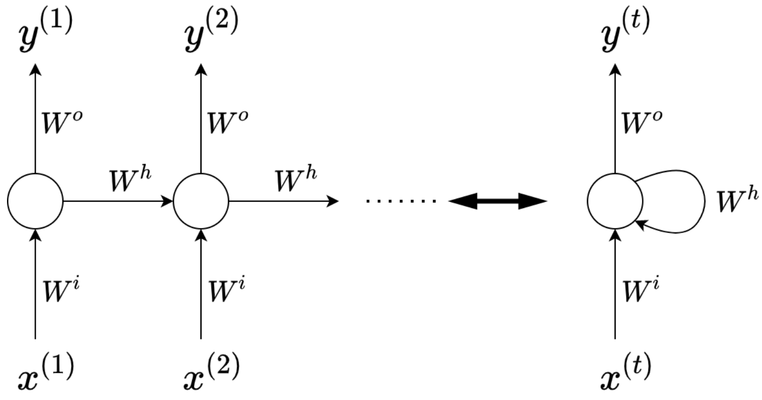


Figure 5.4: Diagram describing the structure of a recurrent neural network architecture, which two architectures on the left and the right are equivalent.

There are variations within RNNs, distinctions include whether the output of the network is produced after every time step or only at the end of the sequence, and whether the recurrent connection to the next time step is made before or after passing through activation functions. A variation in RNN called the gated RNNs are one of the most effective models used in practical applications such as the long short-term memory (LSTM) and networks based on gated recurrent unit (GRU) [50]. Gated RNNs can are formed with cells that replace the hidden neuron unit of ordinary RNNs, which is able to accumulate information in itself that is also able to forget them whenever that becomes redundant with the help of gate units. A gate unit is an unit which sets the weight between the new information and the recurrent information via a sigmoid function, which can be used to control the dependencies of the recurrent information of the input, output and the accumulated information in the self-loop [50].

5.3.3. Training process

The aim of the training process is to tune the parameters $\boldsymbol{\theta}$ of the ANN such that it minimises the cost determined by the cost function of the model $J(\boldsymbol{\theta})$, which can for example be the sum of the Euclidean norms between the ANN outputs and the expected outputs. Typically the weights are tuned using the gradients of the cost function with respect to the weights $\nabla J(\boldsymbol{\theta})$ found using for example using back-propagation for feedforward ANN and back-propagation through time for RNN [50]. The weight can be updated in the $n+1$th iteration using the information from the nth iteration as shown in Equation 5.6, where α is the learning rate.

$$\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}_n - \alpha_n \nabla J(\boldsymbol{\theta}) \tag{5.6}$$

There exists different ways to use the information of the cost J and its gradients to train the model to improve its performance. One example is the stochastic gradient descent, which modifies the weights according to the approximate gradient $\hat{\nabla} J(\boldsymbol{\theta})$ found with randomly selected subset of data [50]. This is an effective alternative to computing the true gradient using the entire dataset at every iteration, which can be very computationally expensive. There exists different design choices for the selection of the learning rate α, which can also change after each iteration designed to speed up the learning process. Examples include the momentum algorithm, root mean squared propagation (RMSProp) and the ADAM optimiser [50].

As discussed earlier, ANN can have a very high approximation power through appropriate choices of model parameters and the training scheme. However this may lead to overfitting, when the parameters are excessively tuned which shows a good performance with the training set but a deteriorated performance with unseen data. This could be for example caused by fitting the model to not only the process but to the noise included in the data. There exists strategies to counteract overfitting through regularisation, for example applying weight penalties to minimise the overall weight of the model and to randomly disregard some neurons in a training to avoid overdependencies on certain neurons [50].

6 An overview of quadrotor drones

In recent years, unmanned aerial vehicles (UAVs) have gained a lot of attention due to their wide range of applications. Among different types of UAVs the quadrotor configuration, the configuration with four independent rotors, is a popular choice. In this section, a brief overview of different applications of quadrotor drones and difficulties faced in their operations are presented. In Section 6.1, a general description of quadrotor drones and their applications are presented. In Section 6.2, different characteristics of this configuration, the difficulties that arise from them are discussed and recent researches to address them.

6.1. The quadrotor configuration

Multirotor configuration is a common configuration for UAVs that have multiple rotors with their rotation axes aligned vertically, which are used for both propulsion and control. This grants them the ability to vertically take-off and to hover in place, which enable them for many different applications. Their applications include reconnaissance, package delivery, agriculture monitoring and filming to name a few [49]. Multirotor drones can be used to enter and to hover in otherwise dangerous areas, for example they can be used to autonomously inspect blades of off-shore wind turbines [54]. Multirotor drones are still seeing much development with various research being performed, while there are also drone racing competitions that push the limits of the capabilities of quadrotor drones [55].

The quadrotor configuration with four individual rotors is popular among different multirotor configurations because of their high manoeuvrability, simplicity in design, light weight, low cost and low maintenance [56]. The quadrotor configuration is inherently unstable [15], which grants high manoeuvrability but makes them more challenging to control as a side effect. Due to its popularity in both practical applications and academic fields, this research project considers the quadrotor configuration among other multirotor configurations.

6.2. Current challenges

Despite the advantages the quadrotor configuration brings, the configuration also has challenges that are still actively being researched today. Emran and Najjaran performed a review on challenges in controlling quadrotor systems in which they categorised them into three: underactuated dynamics, model uncertainties and actuator failure [56]. These challenges are addressed in recent research in light to make their operations safer and more robust, which are also relevant for flight envelope prediction and protection techniques for quadrotor drones.

Underactuated nature

Advantages of the quadrotor configuration such as simplicity in design come at a cost. A quadrotor drone is an underactuated system which means that it has more degrees of freedom to be controlled than the number of actuators that can be independently controlled [56]. In the case of quadrotor drones, the degree of freedom to be controlled are three translational velocities in x, y and z directions (u, v and w) and three rotational rates in x, y and z directions (p, q and r). Unlike the conventional helicopter configuration, the rotors on the quadrotor drones have fixed pitch rate which leaves only the actuators to control the rotor speeds of each rotor as control elements. This makes the number of states to be controlled as six and the number of control inputs as four.

Underactuated nature of the quadrotor configuration highly complicates navigation and stabilisation tasks [56], which are crucial for operation for application examples raised in Subsection 6.1. Hence, different control strategies have been researched in the last decades to effectively control quadrotor drones. Control methods of quadrotor drone range from traditional feedback control, backstepping approach that stabilises the system recursively in a step-by-step manner [57] and incremental nonlinear dynamic inversion that transforms the nonlinear dynamics into a linear input-output map [58]. Artificial intelligence has also been used for quadrotor controls, for example by approximating the nonlinear dynamics and self-tuning of fuzzy PID controller gains [56].

Actuator failures

Another challenge in operation of quadrotor drones exist in the cases of actuator failures, which can make already underactuated, unstable quadrotor drones to be even more difficult to control. However they often perform their tasks close to obstacles or other vehicles, which make them more susceptible to such failures. With one or more of actuator failures, the control strategies for the nominal condition is no longer appropriate to maintain safe flight, for example that it needs to rapidly spin in the yaw direction to maintain altitude.

Identifying the failure during a flight is a challenging task, as well as designing control strategies in different failure modes. Hardware redundancies are often employed that enable them to still operate in some failure modes [56]. Identifying the severity of the failure during operation can be performed with system identification techniques, which are still being developed today for example using advanced variants of Kalman filters [59]. There exist ways to cope with actuator failures in the software side as fault tolerant control. An example is incremental nonlinear dynamic inversion which was shown that it can be used to control quadrotor drones with one actuator failure by Lu and van Kampen [1], or two opposing actuator failures by Sun et al. [2]. When a quadrotor experiences actuator failures, its dynamics change drastically which also affects its flight envelope. A safe state in the nominal flight condition may no longer be safe after actuator failures, which makes flight envelope prediction of quadrotor drones under failure a highly relevant field of research.

Difficulty in modelling the dynamics

Modelling of a system can be highly beneficial in different ways, for example being able to employ advanced control strategies like model based methods and being able to simulate the dynamics of the system without the need for the physical system. Modelling of quadrotor drones has also been a great challenge resulting in discrepancies in the behaviour of the physical drone and its model, called the model gap. For example when a drone is carrying a payload with unknown mass, which may frequently happen during for example package delivery tasks, this causes what is known as parametric uncertainty [60] which is also a form of a model gap. There also exists non-parametric uncertainties, which are model gaps caused by unmodelled nonlinearities like the effects of gust and wind [56]. These model gaps can also have a significant effect on the predicted flight envelope using this model, which may cause large discrepancies between the true and the predicted flight envelopes.

Furthermore, the aerodynamic effects around quadrotor drones are highly complex as shown in researches by for example Foster and Hartman [18] and Sun et al. [49]. Modelling with failure adds another complication, where there may be a need for a different model structure that is more suitable in the off-nominal flight condition, for example while rapidly spinning in the yaw direction as shown by Sun and de Visser [2]. Coupled with the complex aerodynamic effects that can change in different flight conditions, there is a need for more robust or adaptive solutions [56].

7

Dynamic models for investigation

In this chapter, the dynamic systems and their models that could be used to analyse the performance of a flight envelope prediction method are presented. The considered systems vary in their complexities and the number of dimensions, in which the goal is to apply the method to increasingly more complex models and finally to a quadrotor drone model. In Sections 7.1, 7.2 and 7.3 a double integrator problem, an inverted pendulum problem and a quadrotor dynamic model are described, respectively.

7.1. Double integrator problem

A double integrator problem is perhaps the simplest dynamic system, which can also be seen as a "cart on rail" problem. A cart is on a rail which can be pushed to the left or to the right with an external force F. A schematic drawing of the system is shown in Figure 7.1, where it is assumed that the surface is friction-less. The equation of motion of this system is as shown in Equation 7.1, where x_1 is the position of the cart, x_2 is the velocity of the cart, F is the external force in and M is the mass. Now let input u be F/M and define the state vector $\boldsymbol{x} = [x_1\ x_2]^T$ then the system can be written as a state space system as shown with Equation 7.2, where the output vector is the state vector.

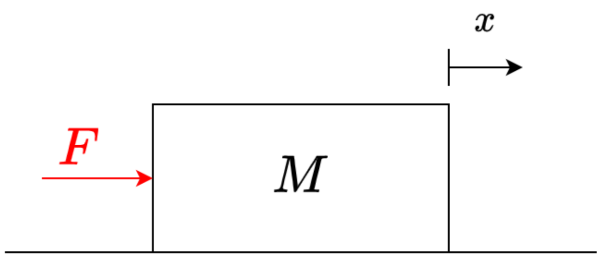


Figure 7.1: Schematic drawing of a double integrator (cart on rail) problem.

$$f(\boldsymbol{x}) = \begin{cases} \dot{x_1} = x_2 \\ \dot{x_2} = \dfrac{F}{M} \end{cases} \tag{7.1}$$

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \tag{7.2}$$

When considering reachability analysis on this system, one of the main advantages is that the system is linear and simple which its motions are intuitive. For instance, one can expect the reachability set to shrink in size when the mass M increases because that makes the mass more resistant to be moved around. Furthermore the system only has two state variables, which makes its reachable set representation to be a more comprehensible, two-dimensional region.

Another advantage of testing reachability analysis methods on this system is that there exists an analytical solution of the reachable sets in terms of the time to reach function starting from the origin with a bounded input space $[-1, 1]$ by Athans and Falb [61]. The analytical value function for the forward $\phi_f(\boldsymbol{x})$ and the backward reachable $\phi_b(\boldsymbol{x})$ sets are shown in Equations 7.3 and 7.4, respectively. Figure 7.2 shows the surface plot (left) and the contours with different time horizons (right) of $\phi_f(\boldsymbol{x})$, together with the switch curve in red which indicates the border in which the optimal control changes its sign [62]. Having access to the analytical solution makes it possible to quantitatively study the accuracy of estimated reachable sets, which is not possible with many nonlinear problems.

$$\phi_f(x) = \begin{cases} -x_2 + \sqrt{4x_1 + 2{x_2}^2} & \text{if} \quad x_1 > \frac{1}{2}x_2|x_2| \\ x_2 + \sqrt{-4x_1 + 2{x_2}^2} & \text{if} \quad x_1 < \frac{1}{2}x_2|x_2| \\ |x_2| & \text{if} \quad x_1 = \frac{1}{2}x_2|x_2| \end{cases} \tag{7.3}$$

$$\phi_b(x) = \begin{cases} x_2 + \sqrt{4x_1 + 2{x_2}^2} & \text{if} \quad x_1 > -\frac{1}{2}x_2|x_2| \\ -x_2 + \sqrt{-4x_1 + 2{x_2}^2} & \text{if} \quad x_1 < -\frac{1}{2}x_2|x_2| \\ |x_2| & \text{if} \quad x_1 = -\frac{1}{2}x_2|x_2| \end{cases} \tag{7.4}$$

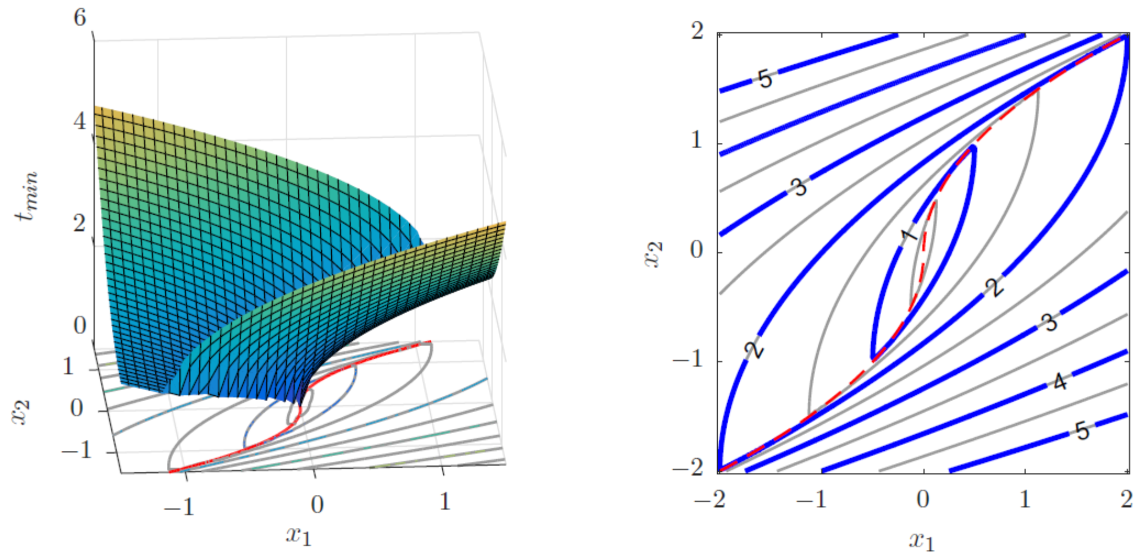


Figure 7.2: The surface plot (left) and the contour plot (right) of the analytical forward reachable set time to reach function of the double integral problem shown in Equation 7.3 [62].

This problem can also be extended by for example including nonlinear terms such as friction force and or aerodynamic drag. It is then possible to compare the reachable set with the original linear problem and to study the effects of nonlinearities on the reachable sets. Therefore this problem is an intuitive problem, which serves as a good starting point to test reachability analysis methods which can be compared to the analytical solutions, while nonlinearities can easily be added to the problem if needed.

7.2. Inverted pendulum problem

An inverted pendulum problem has a simple set up with a complex, nonlinear dynamics. A pendulum with a point mass m is attached to a mass M with a mass-less rod of length l, which a schematic drawing is shown in Figure 7.3. The pendulum is free to swing for any value of angle θ. There is an external force F applied to the system which acts on mass M, where it is assumed that there is no friction between mass M and the surface.

The system has two degrees of freedom, which are the translational motion of mass M and the rotational motion of the pendulum mass m, as shown in the schematic drawing as x and θ respectively. The equation of motion of the pendulum can be derived using Lagrangian mechanics. The kinetic energy E_K and the potential energy E_P of the system can be described as shown in Equations 7.5 and 7.6 respectively.

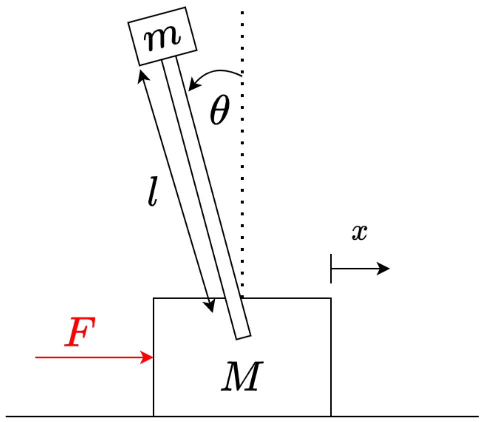


Figure 7.3: Schematic drawing of an inverted pendulum system.

$$E_K = \frac{1}{2}(M+m)\dot{x}^2 - ml\dot{x}\dot{\theta}\cos(\theta) + \frac{1}{2}ml^2\dot{\theta}^2 \tag{7.5}$$

$$E_P = mgl\cos(\theta) \tag{7.6}$$

The generalised coordinates of this system are x and θ, where for a generalised coordinate q and its derivative $\dot{q}$, the equation of motion in that mode can be represented as $\frac{d}{dt}\frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q} = Q$ with the Lagrangian $L = E_K - E_P$, where Q is the generalised force or moment. With this, the equations of motion in the two degree of freedom (translational and rotation) are shown in Equations 7.7. These equations can be reorganised into a set of four ODEs each describing the time derivative of the states $\boldsymbol{x} = [x, \dot{x}, \theta, \dot{\theta}]^T$ in which the non-trivial EOMs are shown in Equations 7.8 and 7.9.

$$\begin{aligned} (M+m)\ddot{x} - ml\ddot{\theta}\cos(\theta) + ml\dot{\theta}^2\sin(\theta) &= F \\ l\ddot{\theta} - \ddot{x}\cos(\theta) &= g\sin(\theta) \end{aligned} \tag{7.7}$$

$$\ddot{x} = \frac{l\ddot{\theta} - g\sin(\theta)}{\cos\theta} \tag{7.8}$$

$$\ddot{\theta} = \frac{F\cos(\theta) + (M+m)g\sin(\theta) - ml\dot{\theta}^2\sin(\theta)\cos(\theta)}{l\big((M+m) - m\cos^2(\theta)\big)} \tag{7.9}$$

The initial condition can greatly affect the behaviour of the motion of the system. For instance, when the system starts with $\theta = \pi$ the system is at its most stable position with the lowest potential energy, while when the system starts at $\theta = 0$ the system quickly converts its high potential energy to kinetic energy even with small disturbances. Although also dependent on the choice of model parameters, the system with former initial condition has relatively smaller effects from the nonlinearities which it can be expected that its reachable sets are similar to those of the double integrator problem discussed in Section 7.1. On the other hand, the reachable sets with the latter initial condition can be expected to be more complex and less intuitive. This also means that it is possible to control the "intensity" of the nonlinearities that can be incorporated in the system dynamics to some extent by adjusting the initial condition and model parameters.

Unlike the double integrator problem, there is no analytical solution of the reachable set for this problem. However this is a widely known nonlinear problem which reachability analysis has been performed in many research. Furthermore, the number of dimension of this problem is four, which makes it possible to compare reachable sets with those computed with the level set method which is still tractable [10].

The inverted pendulum problem has a simple setup where its states are relatively intuitive to grasp, while its dynamics can be complex due to its nonlinearities. Furthermore, the effect of the nonlinearities can be adjusted to some extent with initial conditions and model parameters. Additionally, like the quadrotor dynamics, this is an underactuated system which only has one individual control F with four states. This makes it a good problem to test reachability analysis methods before applying to quadrotor drone models.

7.3. Quadrotor drone dynamics

The main objective of the research is to apply a novel flight envelope prediction technique to quadrotor drones, in which application to a dynamic model of a quadrotor drone serves as the last step of this research. Typically, flight envelopes are predicted separately in longitudinal and lateral modes by using decoupled dynamics to reduce dimensionality. In order to test the flight envelope prediction capability of a method in such cases, testing it on either of them suffices. In this research, a longitudinal dynamic model is considered, which the equations of motion are shown in Equation 7.10 [15], where the states are the longitudinal velocity V_x, vertical velocity V_z, pitch angle θ and pitch rate q in all in the body axis of the quadrotor as depicted in Figure 7.4. m_v is the vehicle mass, I_{yy} is the moment of inertia around the y axis in and g is gravitational acceleration.

$$\begin{aligned}
\dot{V}_x &= \frac{F_x}{m_v} - gsin(\theta) - qV_z \\
\dot{V}_z &= \frac{F_z}{m_v} + gcos(\theta) + qV_x \\
\dot{\theta} &= q \\
\dot{q} &= \frac{M_y}{I_{yy}} \\
\dot{u}_1 &= -\frac{1}{\tau}u_1 + \frac{1}{\tau}u_{1,ref} \\
\dot{u}_2 &= -\frac{1}{\tau}u_2 + \frac{1}{\tau}u_{2,ref}
\end{aligned} \tag{7.10}$$

The aerodynamic forces and moments (F_x, F_z and M_y) are computed using aerodynamic models identified by Sun et al. a stepwise regression method shown in Equation 7.11 where $C_{(\cdot)}$ are the coefficients, which their parameters are shown in Equation 7.12 [15]. u_1 and u_2

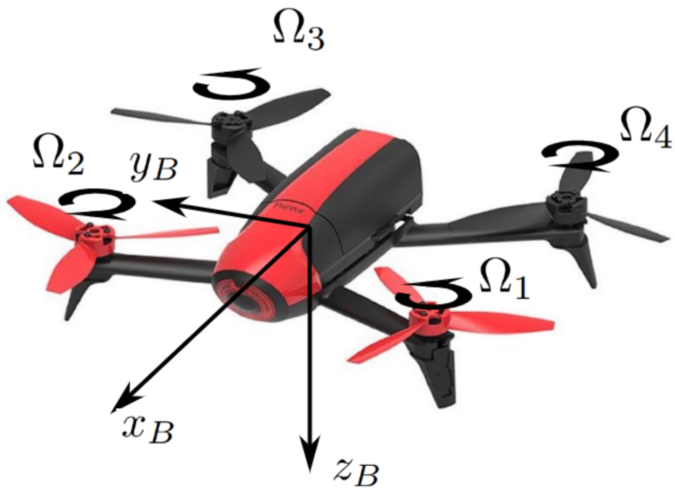


Figure 7.4: The body frame coordinate system and the numbering of rotors of a quadrotor [15].

are to the sum of square of front and back rotor speeds (see Figure 7.4) respectively as shown in Equation 7.13.

$$\begin{aligned} F_x &= C_{d,1}V_x + C_{d,2}V_x^2 + C_{d,3}V_x^3 + C_{d,4}V_z \\ F_z &= C_{z,0} + C_{z,1}u_1 + C_{z,2}u_2 \\ M_y &= C_{m,0} + C_{m,1}u_1 + C_{m,2}u_2 \end{aligned} \tag{7.11}$$

$$\begin{aligned} C_{d,1} &= -2.17 \times 10^{-1} \\ C_{d,2} &= 1.84 \times 10^{-2} \\ C_{d,3} &= -9.61 \times 10^{-4} \\ C_{d,4} &= 6.170460 \times 10^{-2} \\ C_{z,0} &= 2.98 \times 10^{-2} \left|V_z\right| V_z - 3.77 \times 10^{-3} V_z^3 \\ C_{z,1} &= 1.67 - 8.58 \times 10^{-2} V_x + 2.20 \times 10^{-3} V_x^2 \\ C_{z,2} &= 2.15 + 1.97 \times 10^{-2} V_x^2 + 7.28 \times 10^{-2} V_z - 6.84 \times 10^{-4} V_x^3 - 1.97 \times 10^{-4} V_x^3 V_z + 4.34 \times 10^{-3} V_x^2 V_z \\ C_{m,0} &= 1.03 \times 10^{-2} V_x - 6.77 \times 10^{-4} V_x^2 + 8.64 \times 10^{-3} V_z + 7.17 \times 10^{-5} V_x^2 V_z + 2.63 \times 10^{-4} V_x V_z^2 \\ C_{m,1} &= 1.52 \times 10^{-1} + 1.04 \times 10^{-3} V_x^2 + 1.66 \times 10^{-3} V_x V_z - 1.86 \times 10^{-3} V_x \\ C_{m,2} &= -1.63 \times 10^{-1} + 8.04 \times 10^{-3} V_x - 2.11 \times 10^{-4} V_x V_z - 6.31 \times 10^{-4} V_x^2 \end{aligned} \tag{7.12}$$

$$[u_1, u_2] = [\Omega_1^2 + \Omega_2^2, \Omega_3^2 + \Omega_4^2] \tag{7.13}$$

The latter two equations in Equation 7.10 model the rotor actuator dynamics, which their inclusion was shown to have a significant effect on the predicted reachable set as discussed in Chapter 4. τ is the time constant of the actuator dynamics modelled as a first order system mapping the commanded rotor speed u_{ref} to the actual rotor speed u. This makes the system to have six degrees of freedom, making the level set method unsuitable to predict reachable sets. To this date, the only method proved to be tractable for flight envelope estimation of quadrotor drones with actuator dynamics is the MC approach by Sun and de Visser [15]. Hence, developing a faster or more accurate strategy to construct flight envelopes of this problem is the ultimate goal of this research.

Part II

Scientific paper

Trajectory Pruning with Neural Networks for Efficient Monte-Carlo based Quadrotor Flight Envelope Prediction

K. Kaneko * P. Solanki † C.C. de Visser ‡

Delft University of Technology, 2629 HS Delft, The Netherlands

Abstract

Flight envelope prediction is a challenging task where one of the difficulties is that widely used methods, like the level set methods, are impractical for systems with more than four coupled state dimensions due to the 'curse of dimensionality'. Monte-Carlo simulation based approach suffers less from this, however a large number of simulations is needed to predict a flight envelope, while not all simulations directly contribute towards estimating its boundary. This paper proposes a new approach to alleviate this with the use of machine learning techniques that can distinguish more valuable control sequences within the random samples; this knowledge could be used to reduce the number of simulations required to predict the boundary.

An artificial neural network containing a long short-term memory is trained to map a randomly sampled control sequence to the relative position of the resultant end state of the trajectory compared to a predetermined reference reachable set. This trained network is applied for Monte-Carlo based reachability analysis of a dynamic model with model parameter changes compared to the reference model, which is able to reject 50% of randomly sampled sequences while at most 95% of the rejected samples would not have contributed towards reachable set boundary estimation.

1 Introduction

Ensuring the safety of multirotor drones is still one of the most challenging tasks despite their continuous development, which has enabled them to perform increasingly more complicated tasks. One way to enhance operational safety is through an accurate knowledge of system's health and safety in the form of a flight envelope. A flight envelopes is defined as the region in the state space of an aircraft in which the aircraft can operate safely [1]. As long as the aircraft states are within these bounds the aircraft has no risk of entering loss of control in flight (LOC-I), which is the dominant cause of fatal accidents for the past decades [2, 3]. Not only can a flight envelope be used as a reliable, quantitative indication of LOC-I [4], more sophisticated applications of flight envelopes include flight envelope protection. For instance, a flight controller can limit the control inputs to avoid violation of the flight envelope. These techniques are developed for fixed-wing aircraft [5, 6, 7] as well as for multirotor drones [8].

Flight envelopes are typically determined off-line prior to operations. There has been research on different methods for estimating the flight envelope of fixed-wing aircraft. Straightforward methods include wind tunnel testing, physical flight tests and high-fidelity model-based computations [9]. More sophisticated methods include formulating flight envelope prediction as a reachability analysis problem [10], estimating the stability margin through frequency domain analysis [11] and a probabilistic approach with Monte-Carlo simulation [12].

Sun and de Visser [13] have shown that the probabilistic flight envelope prediction approach demonstrated with an unstable fighter jet model by Yin et al. [12] using Monte-Carlo simulation (MCS) can also be applied to quadrotor drones. This allowed prediction of the flight envelope of a longitudinal quadrotor drone model including actuator dynamics, an augmented six-dimensional coupled nonlinear dynamic system, which is greater than the number of dimensions of a tractable problem for the level set approach [8], a widely used reachability analysis method. While the sampling strategy by Sun and de Visser [13] achieved reduction in the samples size of the MCS through tuning a parameter for the sampling process, further reduction may be possible by integrating more sophisticated techniques such as artificial neural networks (ANN). The high prediction power of ANN may allow further reduction of simulations through eliminating superfluous input samples that could not be removed with purely probabilistic approaches.

In this paper, an improved input sampling strategy for MCS-based reachability analysis is proposed that can be used to estimate the flight envelope with a smaller number of numerical simulations, which can be computationally expensive. This is performed by training an ANN to predict the "score" of a simulation output for constructing the reachable set boundary with a given randomly sampled input, which can be used as a reference to circumvent performing unnec-

*M.Sc. Student, Department of Control and Simulation, Faculty of Aerospace Engineering, Kluyverweg 1; k.kaneko@student.tudelft.nl

†Ph.D. Student, Department of Control and Simulation, Faculty of Aerospace Engineering, Kluyverweg 1; p.solanki@tudelft.nl

‡Associate professor, Department of Control and Simulation, Faculty of Aerospace Engineering, Kluyverweg 1; C.C.deVisser@tudelft.nl

essary numerical simulations. First the reachable set of a the system is estimated through a large number of MCS, where each control sequence is assigned a score that describes the relative position of the corresponding trajectory end state with the estimated boundary. An ANN can be trained to map the sampled control sequence to this score, which can serve as a reference to prune control sequences to only perform numerical simulations that results in end state near the reachable set boundary.

This paper is organised as follows. Section 2 describes the flight envelope prediction in a reachability analysis framework using MCS. Section 3 describes the methodology for performing MCS, constructing the reference reachable sets and training the ANNs. Section 4 presents the experimental results together with detailed discussions. Finally, a conclusion is drawn along with future recommendations in Section 5.

2 Safe flight envelope estimation through reachability analysis

In this section, the relationship between flight envelope estimation and reachability analysis is introduced, which is the foundation of the proposed framework of MCS-based reachability analysis. Subsection 2.1 introduces the flight envelope estimation in a reachability analysis framework and Subsection 2.2 describes reachability analysis using MCS.

2.1 Reachable sets

The safe flight envelope is defined as "the region of the aircraft's state space in which the aircraft can be safely controlled and no loss-of-control events can occur" [1]. One of the ways to represent the safe flight envelope is as an intersection between the forward and the backward reachable sets. For a dynamic system $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x})$, the forward reachable set from an initial set K is defined as Equation 1 [10] where $\boldsymbol{\delta}$ is a control sequence in the input domain $\mathcal{U}$, T is the time horizon and $\boldsymbol{\xi}$ is the trajectory. The backward reachable set of the same system can be obtained by analysing the reachable set with the reverse dynamics $\dot{\boldsymbol{x}} = \boldsymbol{f}_{rev}(\boldsymbol{x})$. The reverse dynamics can be described as $\boldsymbol{f}_{rev}(\boldsymbol{x}) = -\boldsymbol{f}(\boldsymbol{x})$ assuming that the system is continuous where there exists a unique trajectory for given initial state, time and input signal [14].

$$\begin{aligned}\mathcal{R}(T, K) = \{\boldsymbol{x} \in \mathbb{R}^n \mid \exists \boldsymbol{\delta}(t) \in \mathcal{U}, \exists \tau \in [0, T], \\ \boldsymbol{\xi}(\tau; t, \boldsymbol{x}, \boldsymbol{\delta}) \in K\}\end{aligned} \tag{1}$$

The intersection of the forward and backward reachable sets represents a set of state were the dynamic system can reach from the initial state within a time horizon T, and also able to return to the initial state within T. By defining the set K as a known safe set and with an appropriate choice of time horizon T, this intersection satisfies the aforementioned definition of the safe flight envelope.

2.2 Reachability analysis using MCS

MCS is one of the methods for estimating the reachable set of a system. MCS is an algorithm used to estimate probability distributions of the system output by calculating outputs using randomly sampled inputs [15]. This can be applied to reachability analysis by defining the input and the output of this system as the control sequence of length T and the state reached at the end of a numerical simulation performed with the said control sequence, respectively. MCS has an important advantage over the level set approach that it can be applied to dynamic systems with a large number of dimensions, where practical application of the level set approach is restricted to dimension of four [8].

In the field of flight envelope prediction MCS has been mainly used for verification and validation purposes [16, 17]. While the end state samples are guaranteed to be contained within the reachable sets assuming no model gap, the MCS-based approach suffers from a problem that a very large number of simulations is needed for accuracy while the required number of samples for a certain level of accuracy remains unknown [18]. Research by Yin et al. [12] and Sun and de Visser [13] both aimed to make reachability analysis with MCS-based approach more efficient, by reducing the sample space using the extreme control effectiveness method [12] and by employing an efficient sampling strategy that has a higher probability of yielding trajectories that travel further from the initial set, respectively.

After a series of numerical simulations, another process is required to formulate a continuous reachable set from the cloud of end state samples. One option is to create an outer contour of the point cloud for example using α-shapes which can be applied in two-dimensional [19] and three-dimensional spaces [20]. α-shape is a polytope formed from a cloud of points, which may not be necessarily convex nor bounded as a single region, in which its extent of non-convexness is defined by the parameter α. α is a real number $0 < \alpha < \infty$, where when $\alpha = \infty$ the resultant α-shape is a convex hull and with decreasing α the corresponding α-shape becomes more non-convex in an incremental manner [20].

Another option is to estimate a probabilistic reachable set described by a multivariate probability density function (MPDF), demonstrated by Yin et al. [12]. The safety of a given state $\boldsymbol{x}$ can be described by the degree of membership of the probabilistic flight envelope $\tilde{E}$ shown in Equation 2. $f_{X_f}(\boldsymbol{x})$ and $f_{X_b}(\boldsymbol{x})$ are MPDF of the random variable X_f and X_b, which are respectively the end states of the forward and backward numerical simulations with randomly sampled control sequences.

$$\mu_{\tilde{E}}(\boldsymbol{x}) = \frac{f_{X_f}(\boldsymbol{x}) \cdot f_{X_b}(\boldsymbol{x})}{\max_{\boldsymbol{x} \in \mathbb{R}^n} f_{X_f}(\boldsymbol{x}) \cdot f_{X_b}(\boldsymbol{x})} \tag{2}$$

Estimation of $f_{X_f}(\boldsymbol{x})$ and $f_{X_b}(\boldsymbol{x})$ can be performed using kernel density estimation (KDE) [21]. Suppose an n dimensional random vector $\boldsymbol{X} = (X_1, X_2, ..., X_n)$ and N samples from it where ith sample is denoted as $\boldsymbol{y}_i = (y_{i,1}, y_{i,2}, ..., y_{i,n})^T$. Then with these samples, the kernel density estimator is shown as in Equation 3. $K_H()$ and $k()$ are the kernel and the normalised kernel function respectively, where h_j are bandwidths. The MPDF are estimated and stored in a grid, which the degree of membership of a given state inside the gridded space can be estimated through interpolation.

$$\begin{aligned} \hat{f}(\boldsymbol{x}) &= \frac{1}{N} \sum_{i=1}^{N} K_H(\boldsymbol{x} - \boldsymbol{y}_i) \\ &= \frac{1}{N \cdot h_1 \cdot h_2 ... h_n} \sum_{i=1}^{N} \prod_{j=1}^{n} k(\frac{x_j - y_{ij}}{h_j}) \end{aligned} \tag{3}$$

3 Methodology

In this section, the methodologies of the experiments performed in this research are presented, which the general framework is described in Figure 1. The experiments performed to train the ANN can be split into four phases as depicted on the left side of Figure 1. First, MCS is performed by producing a large number of random control sequences with a sampling strategy, which are used to simulate the dynamic model to find the state reached at the end of each simulation. These random samples are used to construct a representation of the reachable set, which is used as a reference to assess the value of the trajectory from each simulation for estimating the reference boundary. Finally, an ANN is trained as a trajectory value prediction system that maps a given control sequence to the value of the end state of the corresponding simulation trajectory. This trained ANN can be implemented in MCS-based reachability analysis to reject sampled control sequences which are unlikely to drive the system further towards the boundary of its reachable set, as depicted on the right side of Figure 1.

The experiments are performed with a simple longitudinal dynamic model of a quadrotor drone in hover, which is described in detail in Subsection 3.1. Subsection 3.2 describes the methodology of the MCS process. Subsection 3.3 describes the methodologies of converting the cloud of end state samples from the MCS process into continuous representations of reachable sets. Finally in Subsection 3.4 the ANN architectures considered are described as well as the training methods.

3.1 Dynamic system

In this research, MCS with the ANN integrated to avoid unnecessary simulations is applied to a simplified, longitudinal quadrotor model in hover. The system has four states, which are the horizontal velocity V_x, vertical velocity V_z, pitch angle θ and pitch rate q in all in the body axis of the quadrotor. The equation of motion in the form of the set of ordinary differential equations are shown in Equation 4.

$$\begin{aligned} \dot{V}_x &= \frac{F_x}{m_v} - gsin(\theta) - qV_z \\ \dot{V}_z &= \frac{F_z}{m_v} + gcos(\theta) + qV_x \\ \dot{\theta} &= q \\ \dot{q} &= \frac{M_y}{I_{yy}} \end{aligned} \tag{4}$$

The longitudinal aerodynamic forces and moments during hover are shown in Equation 5, where Ω_i is the ith rotor speed. $i = 1, 2$ correspond to the front rotors and $i = 3, 4$ to the rear rotors. The system model also includes the actuator dynamics for each of the rotors, where the ith rotor speed Ω_i is modelled as first order system mapping from the reference speed $\Omega_{i_{\text{ref}}}$ to actual speed with a constant time constant τ as shown in Equation 6, where the numerical value is shown in Table 1. The numerical values of thrust coefficient during hover κ_0, moment arm l, vehicle mass m_v and mass moment of inertia around y axis I_y are tabulated in Table 1, which are of Parrot Bebop 1 [22, 23].

$$\begin{aligned} F_x &= 0 \\ F_y &= -\kappa_0 \sum \Omega_i^2 \\ M_y &= l\kappa_0(\Omega_1^2 + \Omega_2^2 - \Omega_3^2 - \Omega_4^2) \end{aligned} \tag{5}$$

$$\dot{\Omega}_i = \frac{1}{\tau}(\Omega_{i_{\text{ref}}} - \Omega_i) \tag{6}$$

Table 1: *Parameters of longitudinal quadrotor drone model in hover condition [22, 23] of a Parrot Bebop 1*

m [kg]	I_{yy} [kg m^2]	l [m]	κ_0 [N s^2]	τ [s]
0.389	0.001242	0.0975	1.58e-6	1/30

Since the aerodynamic model used assumes flying in the hover condition, while a fast manoeuvrable system like a quadrotor drone which are likely to deviate from hover even after a brief time, this makes the model less valid with the considered time horizon of 0.15 second. This may cause the reachable sets found with this model to have discrepancies when compared with the true reachable sets of a Parrot Bebop 1. However the main aim of the research is to investigate whether an ANN is able to accurately assess the value of simulating a given trajectory towards constructing the reachable

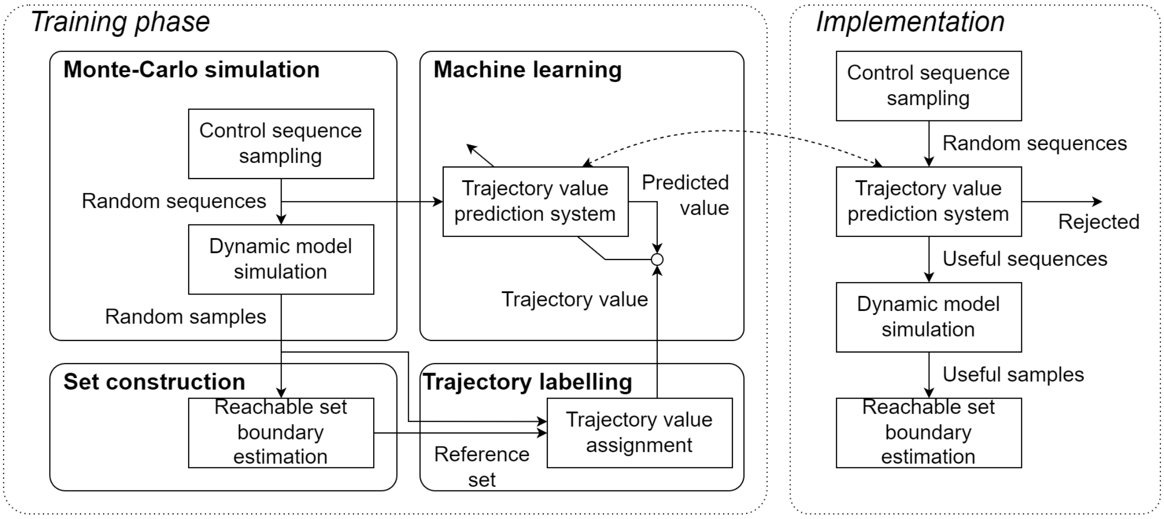


Figure 1: *General framework of the proposed MCS-based reachability analysis with ANN pruning. The left side and the right sides, indicated with dotted boxes, describe the training phase and the expected implementation of the trained ANN respectively.*

set boundary, hence the accuracy of obtained reachable sets themselves is of low importance. The simple aerodynamic model is selected to reduce necessary computation times while maintaining the number of dimensions of the dynamic system.

3.2 Monte-Carlo simulation

The control sequences are sampled with extreme control effectiveness method (ECEM) [12] which only consists of the most extreme control actions in the control space $\mathcal{U}$, while also employing the sampling strategy demonstrated by Sun and de Visser [13]. The strategy makes use of the parameter p_c, which is the probability of sampling a constant sequence, used to control the extent of how extreme the control sequence samples tend to be. It is described as $p_c = (1 - p_s)^N$ where N is the length of the sequence and p_s is the probability of switching the input in ith dimension moving from time k to $k+h$ as described in Equation 7. Since the considered quadrotor drone model's control space is two dimensional, rear and forward rotor speeds, the sequence is sampled twice in each simulation corresponding to each of the control dimension.

$$p_s = 1 - P(u_i(k+h) = u_i(k)) \tag{7}$$

With randomly sampled sequences, the system is simulated through numerical integration of Equation 4. The integration scheme used is the classical 4th order Runge-Kutta scheme. For forward and backward reachable set estimations, the numerical integration can be performed forward in time and backward in time respectively. Numerical integration backwards in time can be performed by multiplying the right hand side of Equation 4 by -1. However when considering actuators modelled as first order systems this cannot be directly applied. When the actual control is very close to the commanded control forward in time the gradient approaches zero until the difference between them are so small they can be treated to be equal. However in this situation reverse in time the knowledge of the past reference control, equivalent to the future when the time propagates forward, is required to accurately simulate the actuator dynamics in such a way the trajectory can be traced forward in time. Hence for the backward simulations the sampled control reference sequences are first flipped in time, to which then the first order lag is applied before flipping the resultant control sequence back in time. This ensured that each of the backward trajectories could be traced forward in time with the same actuator dynamics used for the numerical simulation. However as a side effect, the backward reachable set becomes relatively larger than the forward reachable set because at $t = 0$ the actual control inputs are not necessarily at trim.

After each numerical simulation, the aircraft states reached at the end of the simulation are recorded. These aircraft states are converted from $[V_x, V_z, \theta, q]$ as shown in Equation 4 to $[V, \gamma, \theta, q]$ where V is the airspeed and γ is the flight path angle, which are more suitable for describing the safety of a given aircraft system.

3.3 Reference set construction and score labelling

The end state samples resulting from MCS are synthesised to form a continuous reachable set as a reference, represented as an α-shape or as a MPDF. Then the end state samples are labelled according to the reference, which is assumed as the true reachable set with a large enough sample size. Methodologies for set construction and labelling for each of the representation types are described below.

α-shapes

One of the ways to construct a continuous set from a cloud of points is by generating a boundary by connecting points with lines in two-dimensional spaces and with faces in three-dimensional spaces, for example as a convex hull. However assuming that the reachable set of nonlinear systems can be non-convex [24], convex hull may generate an overestimate of the safe set which is unfavourable in a real-life operation. Hence in this research the α-shape is employed as one of the set representations which can be generated from point clouds that allows the shape to be also non-convex depending on the choice of the parameter α.

While overly convex sets are unfavourable, overly non-convex, underestimated sets are equally unfavourable which can form discontinuities like holes. Therefore a separate end state samples are prepared to judge the extent of under- or overestimation of the set. The α-shape is constructed with 8000 end state samples, with the parameter α selected as the minimum value such that 80% of 2000 end state samples from a separate data set are contained within the resultant α-shape. Using a lower ratio than 80% may cause the resultant α-shapes with very small values of α that results in α-shapes with discontinuities like holes, where higher ratio may result in an almost completely convex α-shape. The end state samples are collected using randomly sampled control sequence as described in Subsection 3.2 with $p_c = 0.3$.

The end states are labelled with a score value described in Equation 8 which reward end states $\boldsymbol{x}_{\text{end}}$ that are closer to the boundary while penalising being too close to the initial state $\boldsymbol{x}_0$. An end state labelled with a smaller S_α is located closer to the reference boundary and hence more valuable as a sample for estimating the reachable set. The distance between a given end state and the boundary is approximated as the distance between the end state and the closest vertex of the α-shape boundary, $\boldsymbol{x}_{\text{boundary}}$. A maximum cap $S_{\alpha_{max}}$ is set to avoid explosions of the score values, which may occur when the end state is located very close to the initial state. $\boldsymbol{x}_{\text{end}}$ labelled with the score value $S_{\alpha_{max}}$ or larger are treated as equally valueless for constructing the reachable set boundary, which this value is set to 10. It is furthermore also possible that $\boldsymbol{x}_{\text{end}}$ lies outside the reference set boundary $\mathcal{R}_{\text{ref}}$, since the reference set is not the ground truth. When this is the case a minus sign is multiplied to the score as shown in the second part of Equation 8.

$$S_\alpha(\boldsymbol{x}_{\text{end}}) = \begin{cases} \min\left(S_{\alpha_{max}}, \frac{|\boldsymbol{x}_{\text{end}} - \boldsymbol{x}_{\text{boundary}}|}{|\boldsymbol{x}_{\text{end}} - \boldsymbol{x}_0|}\right) \\ \qquad \text{if } \boldsymbol{x}_{\text{end}} \in \mathcal{R}_{\text{ref}} \\ -\frac{|\boldsymbol{x}_{\text{end}} - \boldsymbol{x}_{\text{boundary}}|}{|\boldsymbol{x}_{\text{end}} - \boldsymbol{x}_0|} \\ \qquad \text{if } \boldsymbol{x}_{\text{end}} \notin \mathcal{R}_{\text{ref}} \end{cases} \tag{8}$$

Probabilistic flight envelope with KDE

The MPDF representing the probabilistic reachable sets are constructed on a gridded space, for which the domain is set as the smallest hyperrectangle that covers the end state samples expanded by 20% in each dimension around its midpoint. The number of grid cells per dimension is chosen as 31, which may be increased with the cost of increased computation times for both estimation of the MPDF as well as interpolation to compute the membership function. A Gaussian kernel is considered, where the bandwidths are chosen according to Silverman's rule of thumb for a standard multivariate normal density function as described in Equation 9 [21]. For the jth dimension, the bandwidth h_j depends on the standard deviation σ_j, the number of dimensions d and the number of samples N.

$$h_j = \sigma_j \left[\frac{4}{(d+2)N}\right]^{\frac{1}{d+4}} \tag{9}$$

The probabilistic reachable sets are estimated using KDE, with 10000 MCS end state samples collected with a lower value of $p_c = 0.001$ compared to samples collected for α-shapes. This is to ensure that the states around the initial state are considered 'safe', which is determined by the degree of membership of these states. When a larger value of p_c is used the sampled control sequences are relatively more aggressive leading to less samples around the initial state, resulting in a probabilistic reachable set where the trim condition is also considered relatively 'unsafe'.

When labelling the end states according to the probabilistic reachable set, the degree of membership shown in Equation 10 itself can serve as an indication of how close the end state is towards the boundary. Although this representation of the reachable set does not have an explicit definition of the boundary, the membership function μ_{f_X} has an appropriate characteristic such that the smaller the degree of membership, the further into the state space the end state lies from the initial state. Since the MPDF f_X is computed in a grid, the degree of membership of a state inside the domain is interpolated linearly.

$$S_{\text{KDE}}(\boldsymbol{x}_{\text{end}}) = \mu_{f_X}(\boldsymbol{x}_{\text{end}}) = \frac{f_X(\boldsymbol{x}_{\text{end}})}{\max_{\boldsymbol{x}\in\mathbb{R}^n} f_X(\boldsymbol{x}_{\text{end}})} \tag{10}$$

3.4 Artificial Neural network training

In this subsection, the methodology of ANN construction and training are described. First two types of ANN architectures are introduced, which are feedforward neural networks and long short-term memory networks. Next, the details of the considered architectures and their training processes are described.

Feedforward neural network

The feedforward neural network (FFNN) is the simplest form of ANN, which typically consists of an input layer, an output layer and hidden layers. The neurons in each layer are densely connected to neighbouring neurons with corresponding weights, which is mathematically shown in Equation 11. $\boldsymbol{z}$, $\boldsymbol{v}$ and $\boldsymbol{y}$ are outputs of the input, hidden and output layers respectively. W and b are weight vectors and the bias respectively, where the superscripts h and o denote hidden and output layers respectively. The input vector $\boldsymbol{x}$ enters the network through the input layer, which flows to densely connected hidden layer neurons. $\boldsymbol{z}$, the weighted sum together with the bias passes through the activation function σ and exists the hidden layer as $\boldsymbol{v}$. $\boldsymbol{v}$ then flows to the densely connected output layer neurons, which their weighted sum is computed as the network output $\boldsymbol{y}$.

$$\begin{aligned} \boldsymbol{z} &= W^h \boldsymbol{x} + b^h \\ \boldsymbol{v} &= \sigma(\boldsymbol{z}) \\ \boldsymbol{y} &= W^o \boldsymbol{x} + b^o \end{aligned} \tag{11}$$

Long short-term memory

Recurrent neural networks are a type of network that are able to accept sequences of inputs, where the information is fed into the network in a recurrent manner. A common disadvantage of recurrent network architectures is that the internal states may either grow or shrink indefinitely as they flow through the system. Long short-term memory (LSTM) with forget gates are able to adaptively forget internal states preventing them to explode [25].

LSTM networks contain LSTM cells, in which the mathematical processes are described in Equation 12, where $\odot$ denotes element-wise multiplication, σ_{act} is the activation function and σ_{gate} is the gate activation function. An LSTM cell takes in the input of the current time step, the output of the LSTM cell from the previous time step and the internal state denoted as $\boldsymbol{x}$, $\boldsymbol{h}$ and $\boldsymbol{s}$ respectively. Each cell contains an input gate, an output gate and a forget gate which their outputs are denotes as g, q and f respectively. Each gate is characterised by the input weights U and the recurrent weights W, which passes through σ_{gate}, which is typically a sigmoid function bounding their outputs in [0,1]. These values determine how much of the corresponding information is used to compute $\boldsymbol{h}$ and $\boldsymbol{s}$.

$$\begin{aligned} h^{(t)} &= \sigma_{act}(s^{(t)}) \odot q^{(t)} \\ s^{(t)} &= f^{(t)} \odot s^{(t-1)} + g^{(t)} \odot \tilde{s}^{(t)} \\ \tilde{s}^{(t)} &= \sigma_{act}(b + U\boldsymbol{x} + W\boldsymbol{h}) \\ g^{(t)} &= \sigma_{gate}(b^g + U^g \boldsymbol{x} + W^g \boldsymbol{h}) \\ f^{(t)} &= \sigma_{gate}(b^f + U^f \boldsymbol{x} + W^f \boldsymbol{h}) \\ q^{(t)} &= \sigma_{gate}(b^o + U^o \boldsymbol{x} + W^o \boldsymbol{h}) \end{aligned} \tag{12}$$

Training ANN for trajectory pruning in MCS

In the MCS process, in order to only simulate sampled inputs that result in a valuable end state to construct the reachable set, an ANN is considered to predict the score values introduced in Subsection 3.3 of resultant end state for a given control sequence sample. Four networks are trained through a supervised learning procedure with the control sequence and the labelled score of the corresponding end state as the input and the output, respectively. Two FFNN are trained with the training data sets collected using α-shape as the reference and using MPDF as the reference. Similarly, two LSTM networks are trained with these separate training data sets.

Training input and output data are prepared with the following procedure. First MCS is performed with the quadrotor model from Subsection 3.1 for which the input samples are randomly generated with $p_c = 0.1$. The resulting end states are labelled according to Equations 8 or 10 with the reference sets described in Subsection 3.3. The scores S_α and S_{KDE} are normalised by dividing them by the maximum value observed in the prepared data set. The sampled control sequence and the normalised scores are the input and the output data for the neural networks. 6400 input-output pairs are collected, in which 80% of them are used for training while the remaining pairs are used for validation.

The architecture of the FFNN considered is as follows. The network consists of an input layer, an output layer and one hidden layer. The number of hidden neurons is set to 32 with a *tanh* activation function. The input of the FFNN is control sequence in each of the two control space, which are flattened from $N_{\text{traj}} \times 2$ to $2N_{\text{traj}} \times 1$ where N_{traj} is the length of the control sequence. The output of the network is a scalar value, which is the normalised value of S_α or S_{KDE}. The input and the hidden layers use biases, resulting in a total number of trainable parameters of 6465. The loss function is selected as the mean squared error (MSE) where an ADAM optimiser is applied, in which training is performed with a batch size of 128 which is shuffled in every epoch with 100 epochs.

Another architecture considered has a similar structure as FFNN, where the hidden layer is replaced by a LSTM cell. The number of neurons of the hidden layer and the gates are all selected as 32, where the gates are a sigmoid function and the activation function is a *tanh* function. Unlike FFNN, LSTM cells can take in sequences of inputs. Hence the control sequences are fed into the network directly as a two-dimensional vector in each time step without being flattened first. Each of the gates, the hidden layer for the internal state as well as the LSTM cell output use biases, resulting in a total of 4513 trainable parameters. Similarly to FFNN, LSTM networks are trained to optimise the MSE of the predicted scalar value with the ADAM optimisation scheme, with 200 epochs with a batch size of 128 which is shuffled in every epoch. Since both the training and validation data are collected through the same random sampling strategy, there is a possibility that there are duplicates among these samples. However the control sequences for the forward and the rear rotors were sampled independently, hence the probability that there exist duplicates in the sampled data sets are assumed to be negligible.

The number of computations required to analyse a given input to predict the output using a LSTM is proportional to the length of the input sequence fed into the network. This is a favourable characteristic when using this network to act as a filter to reduce the number of numerical simulations of a dynamic model. This is because while the number of time steps for a numerical integration is bounded to ensure stability, the length of the input fed into the LSTM is that of a commanded control sequence which can be chosen arbitrarily. Using a more advanced, nonlinear system like the quadrotor drone may require a very short time steps for numerical integration. Therefore if LSTM networks are able to accurately predict the value of each of the control sequences, they may enable both reduction in the number of simulations as well as reduction in the overall computation time.

4 Results and discussions

In this section the results obtained during the experiments described in Section 3 are presented together with their discussions. The results of construction of the reference reachable sets are presented in Subsection 4.1. The verification results of the scores labelled to the end state samples are presented in Subsection 4.2. The performance of the trained ANN are presented in Subsection 4.3. The system performance of integrating the trained ANN into a MCS-based reachability analysis is presented and discussed in Subsection 4.4. Finally, the validation results using off-nominal dynamic models are presented in Subsection 4.5.

4.1 Reference set and value assignment

Before performing MCS on the quadrotor drone model, verification is performed using a double integrator system shown in Equation 13. The states variables are x_1 and x_2, with the input u. With the input range $u \in [-1, 1]$ and the initial state of $x_1 = 0, x_2 = 0$, the analytical expression of the reachable set for a given time horizon T is $\phi = T$, where ϕ is shown in Equation 14 [26].

$$\begin{aligned} \dot{x_1} &= x_2 \\ \dot{x_2} &= u \end{aligned} \tag{13}$$

$$\phi(x) = \begin{cases} -x_2 + \sqrt{4x_1 + 2{x_2}^2} & \text{if} \quad x_1 > \frac{1}{2}x_2|x_2| \\ x_2 + \sqrt{-4x_1 + 2{x_2}^2} & \text{if} \quad x_1 < \frac{1}{2}x_2|x_2| \\ |x_2| & \text{if} \quad x_1 = \frac{1}{2}x_2|x_2| \end{cases} \tag{14}$$

Figure 2 shows 3000 end states after simulating the double integrator system with randomly sampled control sequences of lengths of 1 second plotted in blue, together with the isocontour of $\phi = 1$ shown in Equation 14. The control sequences are sampled as bang-bang controls, using the sampling strategy by Sun and de Visser [13]. It can be observed that the cloud of end state samples converges to form the theoretical reachable set of the system. This suggests that MCS is an effective reachability analysis approach given a large enough sample size.

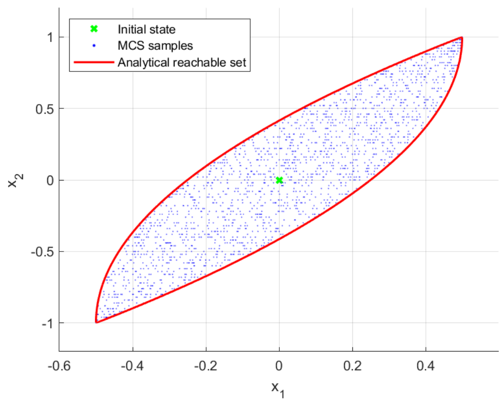


Figure 2: *States reached at the end of simulating the double integrator system with randomly sampled control sequences with the lengths of 1 second (blue) starting from the initial state (green) plotted together with the analytical reachable set (red).*

Figure 3 shows the end state samples of MCS for two of the three-dimensional slices in the state space, $V-\gamma-\theta$ and $\gamma - \theta - q$. Each point indicates the state reached starting from the initial state being the hover condition ($V = 0, \gamma = 0, \theta = 0, q = 0$) shown in green, after forward and backward simulations shown in blue and red respectively. 3000 samples are shown for both the forward and backward end states, with the input control sequence were sampled with $p_c = 0.1$. The point clouds are forming V-like shapes, that are most likely converging towards the theoretical reachable sets as observed with the double integrator system. The point clouds include very extreme states with large deviations from the initial state. For example the forward reachable set ranges from about -2.5 rad to 2.5 in θ and from about -3 rad to 4.5 rad in γ. These states are far from the hover condition which is an assumption made with the dynamic model shown in Equation 4, although the validity of the model itself is of less importance in this research. It can also be observed that the region in which the backward simulations can cover is relatively larger than the forward simulations. This is most likely because in the process of applying the actuator dynamics reversed in time, the actual input is not necessarily at the trim control at the initial condition.

From the point clouds, the forward reachable sets are presented as an α-shape in Figure 4 and as an isocontour of the MPDF estimated with KDE in Figure 5. The isocontour of $k\sigma$ corresponds to $\mu(\boldsymbol{x}) = e^{-k^2/2}$. The backward reachable sets can also be represented in a similar manner, using the point cloud from the backward simulations. It can be seen that both representations share similar shapes already visible in Figure 3.

Representing the reachable set as a MPDF has an added benefit over the α-shape that the reachable set can be made in higher-dimensional spaces. The plot shown in Figure 5 are isocontours of three-dimensional projection of a four-dimensional PDF, these dimensions being the longitudinal states $[V\ \gamma\ \theta\ q]$. While α-shapes are constructed with simulations with all of these states into consideration, the safety of a given state cannot be guaranteed even if it lies within the α-shape in one of the projections as the state not shown in this space may be outside the safe region. Furthermore, a MPDF is also able to describe the level of safety which can also be seen from Figure 5. The inner shapes are isocontours with 2σ while the outer shapes are isocontours with 3σ, meaning that the states inside the inner contours are relatively safer than that inside the outer contours.

However the α-shapes can be theoretically be drawn with a smaller number of end state samples which lie on the surface of the boundary. For example, the α-shapes shown in $V-\gamma-\theta$ and $\gamma-\theta-q$ have 2170 and 2620 sample points used to construct the boundary itself, while in both cases 8000 samples are available. On the other hand reproducing the MPDF with a reduced number of samples most likely results in degradation of the accuracy. Furthermore sampling strategies that favour excessively aggressive control sequences, which may be beneficial for constructing the boundary to sample more end states that reach further out into the state space, is not suitable for MPDF construction. This may result in an envelope where the region around the a-priori known safe set, possibly the safe set itself, to be considered 'unsafe' which is not an accurate representation of the reachable set hence the safe flight envelope. Having to use a more modest sampling strategy results in an increasing number of simulations needed to collect end state samples which are possible to reach but relatively more difficult to reach.

4.2 Verification of the score labels

The labelled scores of the end state sample points are verified by constructing convex hulls in three-dimensional projections of the state space using different selections of end state samples depending on their labelled scores. A convex hull has the largest volume within the family of polytopes that can be formed with a given point cloud, therefore these volumes can serve as the upper bounds of the volumes of the reachable sets. It can be expected that the end states with relatively better scores are able to form convex hulls with larger volumes, that encapsulates convex hulls formed with end state samples with worse scores. An example is shown in Figure 6, where the best 20% and the worst 20% out of 10000 end state samples according to the score labelled using Equation 8 are plotted in green and in red respectively. The contours in lighter blue and in darker blue are convex hulls formed with respective groups of end state samples. While both convex hulls are formed with the same number of samples of 2000, collected with the same MCS sampling strategy, green end state samples form a relatively larger contour than that formed with the red samples.

Considering the scores labelled according to the an α-shape shown in Figure 6, there exists a small portion in the state space which the darker convex hull covers while the lighter convex hull does not. This can be observed in for example the $V-\gamma-\theta$ projection shown in Figure 6(a) with relatively higher values of V. This may be caused by the fact that while the states are four dimensional, the scoring results shown in Figure 6 were made with an α-shape in $\gamma-\theta-q$. On the other hand, such observation cannot be made in $\gamma-\theta-q$ projection shown in Figure 6(b), where the darker convex hull is completely encapsulated in the lighter one. However in general the volume covered by the lighter convex hull is relatively larger. This suggests that the score labelling system shown in Equation 8 rewards samples further out in the state space, which is favourable in collecting samples that are more likely to reach the boundary of the reachable set.

A similar set of plots are shown in Figure 7, where the end state sample points labelled according to Equation 10 with a MPDF as a reference reachable set. A similar trend can be observed that the end state points labelled with better scores form larger convex hulls compared to those with worse scores. However a relatively large number of end state points labelled with worse scores lie outside the convex hull formed with the best samples, in the region with large values of γ as can be observed in Figures 7(b) and 7(c). This reflects how the scores are labelled according to Equation 10 where the score represents how likely a given end state can be reached not necessarily its relative position in the state space from the initial state. The projected space $V-\gamma-\theta$ is bounded in dimension V as it cannot be smaller than zero, which results in a space where the boundary of the reachable set is not necessarily unlikely to be reached, for example the initial state also lies on the boundary of the reachable set. This suggests that more samples are needed than the best 20% of the MCS samples when using the scoring

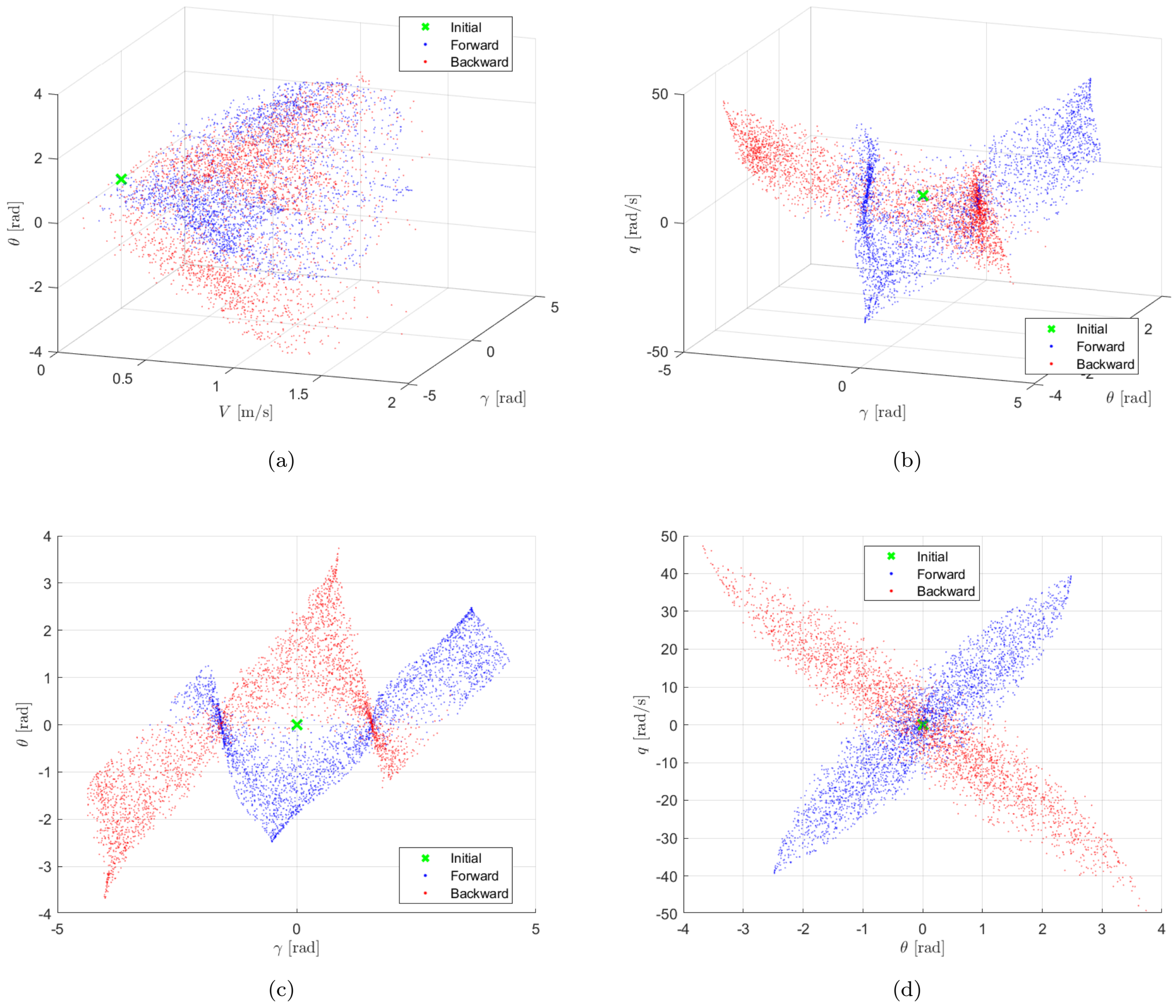


Figure 3: *States reached at the end of forward (blue) and backward (red) simulations of 0.15 seconds of a longitudinal quadrotor drone model from initial condition (green) with randomly sampled control sequences plotted in: (a) $V-\gamma-\theta$ (b) $\gamma-\theta-q$ (c) $\gamma-\theta$ and (d) $\theta-q$. 3000 end state samples are shown for both the forward and the backward reachable end states.*

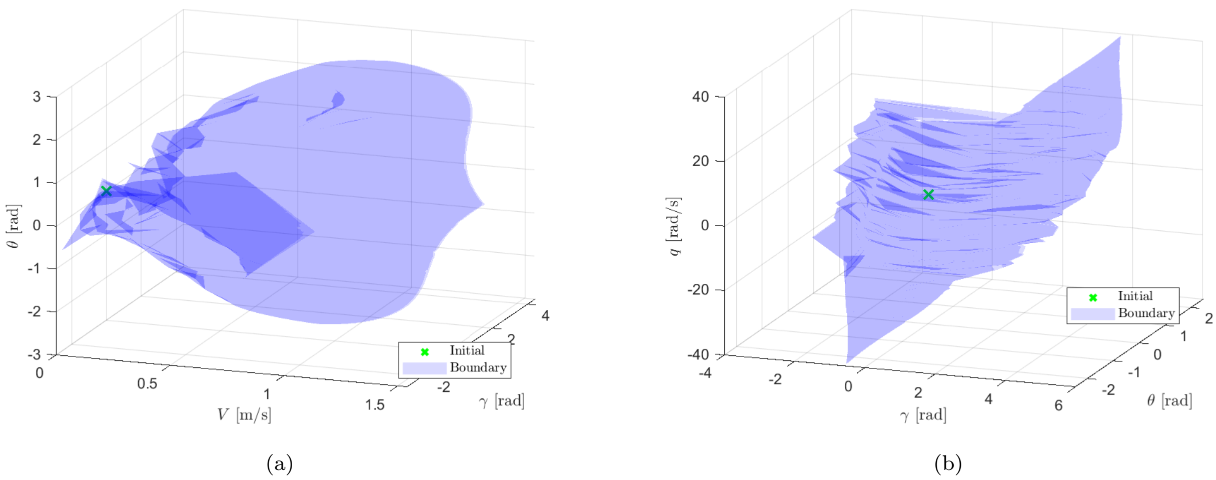
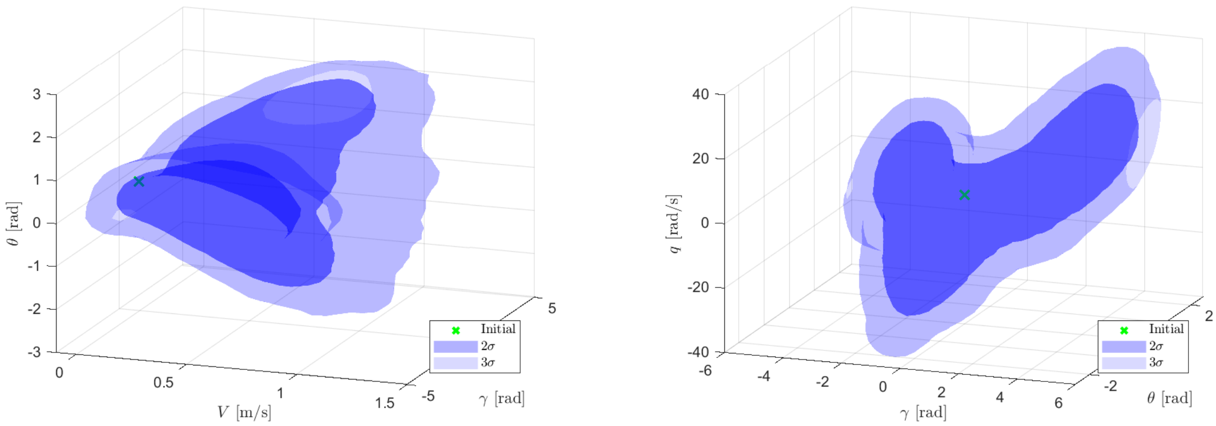


Figure 4: *Reachable set boundary represented as α-shapes using end state samples from MCS (blue) from the initial state at hover (green) in (a) $V-\gamma-\theta$ and (b) $\gamma-\theta-q$ projections.*



(a) (b)

Figure 5: *Isocontours of the reachable set represented as MPDF using end state samples from MCS (blue) from the initial state at hover (green) in (a) $V-\gamma-\theta$ and (b) $\gamma-\theta-q$ projections. The darker inner hull is the isocontour of 2σ and the lighter hull is of 3σ.*

system shown in Equation 10 in order to reconstruct the boundary of the reachable set.

Table 2 shows the verification results of the scores labelled on each of the end state samples, which indicate how valuable the given end state is towards estimating the reachable set boundary. The values in the table indicate the volumes of the three-dimensional convex hulls formulated with end state samples labelled with the scores in the corresponding percentiles. The left two columns describe which of the three-dimensional projections the convex hulls are constructed in and the type of the reachable set representation used as the reference in order to label the end state samples. Each column corresponds to a selected percentile of 10000 sampled MCS end state points, when their labelled scores in ascending order. For example when collecting end state samples with between the lowest and the 2000th lowest scores in the 10000 samples (0-20 percentile) according to Equation 8 using an α-shape as a reference, they formulate a convex hull in the $V-\gamma-\theta$ projection with the volume of 15.5 $[m/s\ rad^2]$.

It can be observed that with the end states in lower percentiles, which correspond to lower scores hence closer to the reference set boundaries, the convex hulls formed have larger volumes compared to those formed with higher percentile end state points. For example convex hulls with the best 20% (0-20 percentile) and the worst 20% (80-100 percentile) of end state samples, which are shown in Figures 6 and 7, the ratios between their volumes can be up to 5. It can be seen that for both $V-\gamma-\theta$ and $\gamma-\theta-q$ projections the end states with the scores in the 0-20 percentile according to the α-shape covers 94% and 100% of the volumes in the respective projections made with all of the end state samples. On the other hand, with the MPDF the volume covered with the same number of samples have volumes of about 87% and 86%. However the scoring system using MPDF shows comparable performance with that with α-shape when considering the percentile of 0-40 in the $V-\gamma-\theta$ projection. This agrees with the observation made from Figure 7 that when using the scores according to Equation 10 with MPDF as the reference set, relatively more samples may be needed to replicate the boundary compared to the scores according to Equation 8 with an α-shape as a reference.

Furthermore it can be observed that the increase in the volumes of the convex hull with additional end state samples with relatively worse scores, for example comparing the volumes with 0-40 percentile and 0-80 percentile which has double the amount of points, is limited. This means that a large portion of the end state samples with relatively worse scores are contained within the convex hull formed with the better scoring samples, suggesting that they are less likely to contribute towards constructing the reachable set boundary.

These results suggest that the scores using both α-shapes and MPDF, shown in Equations 8 and 10 respectively, are in general able to describe the position of corresponding end states relative to the reference reachable set boundary as well as the initial state, favouring samples closer to the boundary and further away from the initial state as intended.

4.3 Neural network performance

Table 3 summarises the prediction performance of the trained ANNs as described in Section 3, which tabulates the mean squared error (MSE), the standard deviation of the squared error (std SE) and the cross-correlation between the predicted and the expected

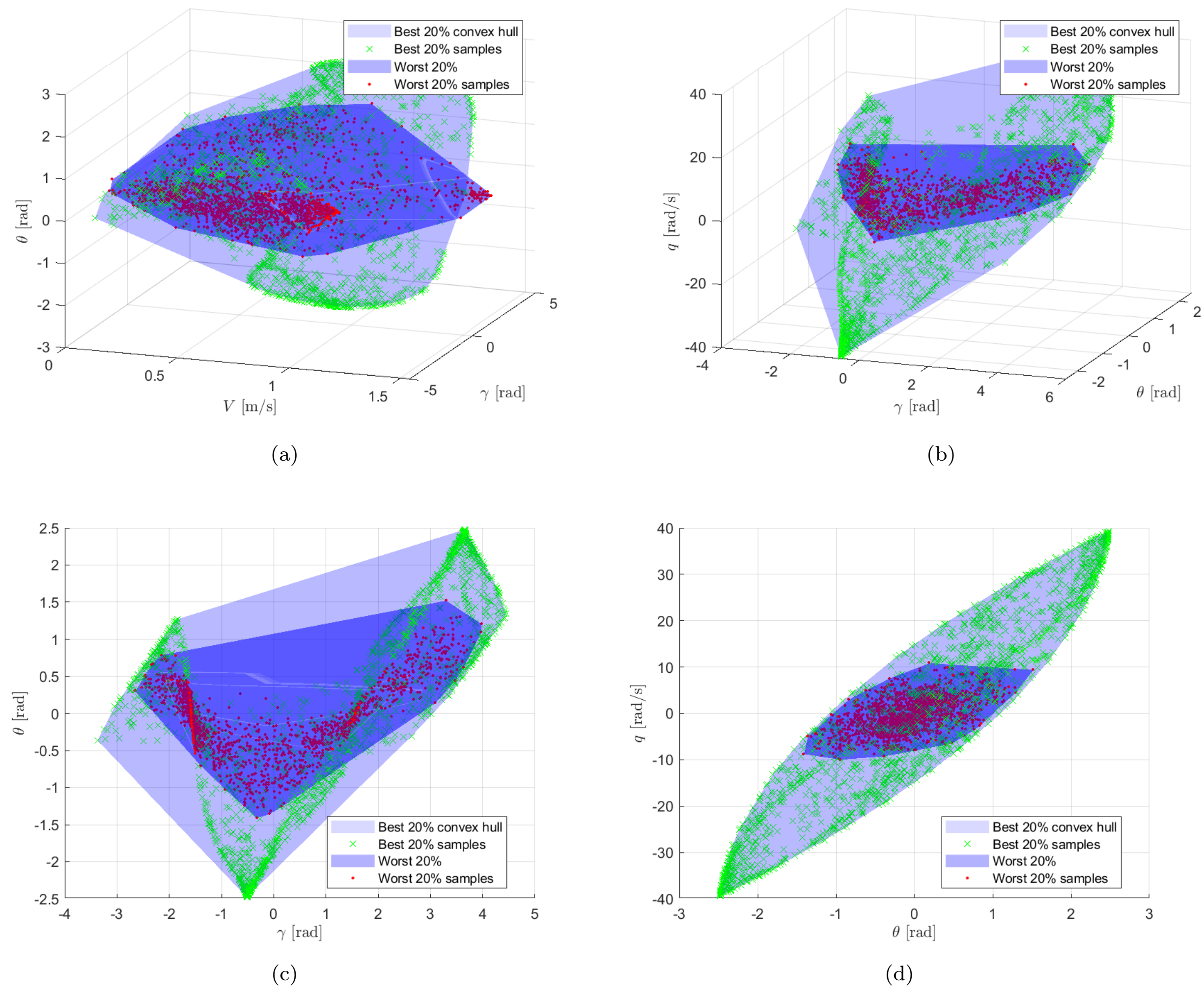


Figure 6: *Top 20% (green) and bottom 20% of 10000 end states of MCS on quadrotor drone system according to the scoring system in Equation 8 with an α-shape in $\gamma - \theta - q$ as the reference, together with convex hulls formed with respective samples (darker blue and lighter blue). Projections in (a) $V - \gamma - \theta$ space (b) $\gamma - \theta - q$ space (c) in $\gamma - \theta$ space (d) in $\theta - q$ space.*

Table 2: *The volumes of the convex hull constructed in respective three-dimensional slices of the state space with MCS end states labelled with scores according to Equations 10 or 8, with scores in the corresponding percentiles when sorted in an ascending order.*

Projected space	Reference type	Percentile									
		0-20	20-40	40-60	60-80	80-100	0-20	0-40	0-60	0-80	0-100
$V - \gamma - \theta$	α-shape	16.7	14.1	13.3	11.2	8.7	16.7	16.9	17.1	17.4	17.7
	MPDF	15.4	13.6	10.8	8.1	5.2	15.4	17.4	17.6	17.7	17.7
$\gamma - \theta - q$	α-shape	444.0	314.4	308.6	239.2	127.7	444.0	444.0	444.0	444.0	444.0
	MPDF	382.1	418.9	334.5	247.0	166.7	382.1	439.0	443.8	444.0	444.0

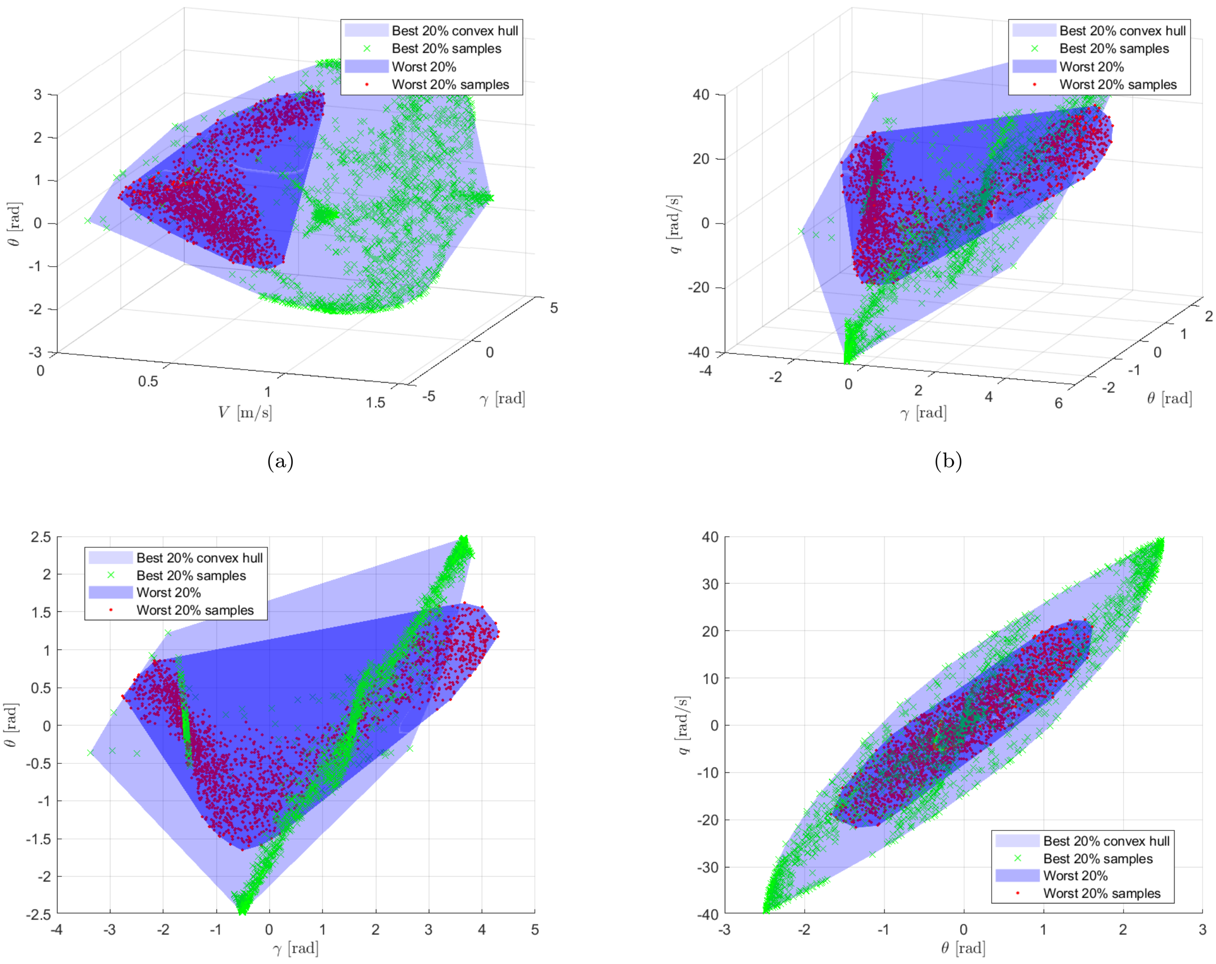


Figure 7: *Top 20% (green) and bottom 20% of 10000 end states of MCS on quadrotor drone system according to the scoring system in Equation 10 with a MPDF as the reference, together with convex hulls formed with respective samples (darker blue and lighter blue). Projections in (a) $V-\gamma-\theta$ space (b) $\gamma-\theta-q$ space (c) in $\gamma-\theta$ space (d) in $\theta-q$ space.*

outputs together with the score range indicating the minimum and the maximum values of the expected outputs. The percentages inside the parentheses describe the relative values compared to the difference between the maximum and the minimum scores of the corresponding data set. All the values are computed with the validation data sets generated with the same procedures as the training data but are not used during training. The different ANN models are characterised by following factors:

- The type of ANN used, either FFNN or LSTM.
- The reference set construction method, either as α-shape or as MPDF.
- The state (sub)space which the reference sets are constructed in, as α-shapes are restricted to three-dimensional spaces.

It can be observed that in general LSTM models have higher prediction power compared to the FFNN counterparts, with lower MSE and higher correlations. This is especially true when considering the reference reachable set as a MPDF, where the FFNN yields a MSE three times larger than that of the LSTM, while also being more consistent which can be seen from the lower std SE value.

There exist discrepancies in the performance of the ANN models with the type of set representation of the reference reachable set among the same type of ANN architecture. This is most evident with the cross-correlation values, where the ANNs trained with the MPDF show higher correlation than the ANNs trained with α-shapes.

Furthermore, the performance of the ANNs depends on the selected three-dimensional space the reference α-shape is constructed. Both the FFNN and LSTM trained with α-shapes in $V-\gamma-\theta$ relatively larger MSE and std SE than that with α-shape in $\gamma-\theta-q$, suggesting they are less accurate and more inconsistent. These discrepancies are most likely caused by the difference in the distributions of the scores described using respective reference sets. The definition of the scores with an α-shape as a reference shown in Equation 8 results in a larger range in the normalised score including more extreme values when the end state is closer to the initial states even with the score caps applied. Which is more prominent with α-shape in $V-\gamma-\theta$ which yielded a score as low as -4.62, suggesting that there were end state samples outside the reference boundary relatively further away from both the boundary and the initial state. Such data may negatively influence the prediction power of the ANN, either resulting in large errors when trying to predict these extreme outputs or globally skewing the model when training with these data points.

No explicit regularisation was applied to the networks such as dropouts, however both the histories of training and validation losses were tracked throughout the training process. Figure 8 shows the histories of the training losses and the validation losses against the number of epochs during training for each of the ANN models considered. The top row (8(a), 8(b) and 8(c)) presents the loss histories of the FFNNs, while the bottom row (8(d), 8(e) and 8(f)) presents the loss histories of the LSTMs. Each row corresponds to the type of reference set and the state (sub)space they are defined in; the left column corresponds with α-shape in $V-\gamma-q$, the middle column with α-shape in $\gamma-\theta-q$ and the right column with MPDF in $V-\gamma-\theta-q$.

Some evidence of overfitting can be observed in some of the trained ANNs where there are larger discrepancies between the training and validation losses, for example in Figures 8(a) and 8(d) which correspond to FFNN and LSTM trained with α-shape in $V-\gamma-\theta$, respectively. Similar observation can be made for Figure 8(e) which corresponds to LSTM trained with α-shape in $\gamma-\theta-q$, but to a lesser extent. These models may benefit more from regularisations in order to generalise the models. On the other hand the remaining ANN models are less susceptible to overfitting, where with an appropriate choice of the number of epochs the gaps between the training and the validation losses can be reduced.

In summary, for the application of pruning control samples of MCS-based reachability analysis of a dynamic system, LSTM is more suitable than FFNN with its higher prediction power observed from both the MSE and the correlation between the prediction and the reference data. Considering the type of representation for the reference reachable set, the MPDF described in the full four-dimensional state space results in more accurate and consistent predictions, also showing relatively less signs of overfitting. Another candidate is the LSTM using α-shape described in $\gamma-\theta-q$ projection with comparable prediction power as with the MPDF, although showing some signs of overfitting.

4.4 System performance with nominal model

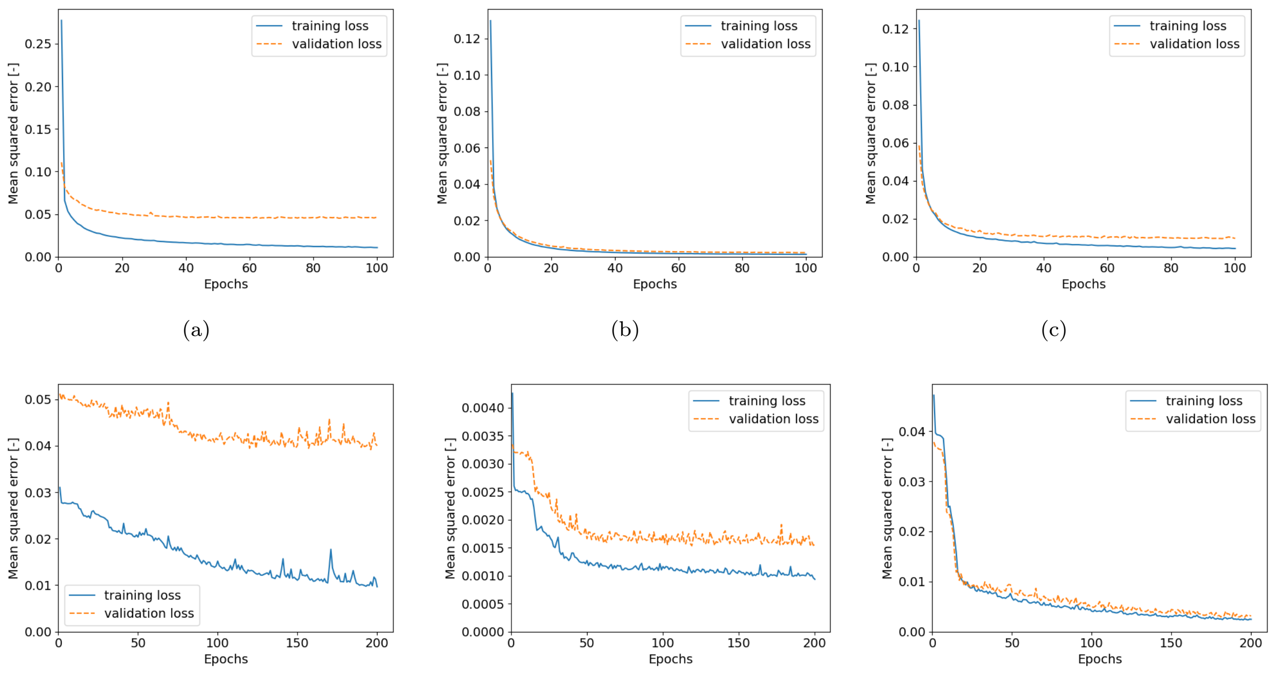
Based on the results shown in Subsection 4.3, following two ANN models are analysed further for their performance when integrated in the reachability analysis using MCS.

- LSTM with MPDF in $V-\gamma-\theta-q$ as the reference set
- LSTM with α-shape in $\gamma-\theta-q$ as the reference set

Figures 9 and 10 show the results of MCS represented as scatter plots, where each point represents a state reached at the end of a forward numerical simulation performed on the dynamic model with a randomly sampled control sequence. The points are labelled with different colours depending on the predicted score with

Table 3: *Performance analysis result of the ANNs trained to map the commanded control sequence to the score of the corresponding end state, presented as mean squared error (MSE), standard deviation of the squared error (std SE) and the cross-correlations between the reference outputs and the predicted outputs together with the minimum and the maximum values of the expected scores in the data set.*

ANN type	Reference set type	Projected space	MSE (relative to score range)	std SE (relative to score range)	Correlation	Score range
FFNN	α-shape	$[V\ \gamma\ \theta]$	3.01E-02 (0.5%)	4.32E-01 (7.7%)	0.50	[-4.62, 1]
FFNN	α-shape	$[\gamma\ \theta\ q]$	3.43E-03 (0.3%)	3.24E-02 (2.4%)	0.48	[-0.34, 1]
FFNN	MPDF	$[V\ \gamma\ \theta\ q]$	9.29E-03 (0.9%)	2.07E-02 (2.1%)	0.87	[0, 1]
LSTM	α-shape	$[V\ \gamma\ \theta]$	2.92E-02 (0.5%)	4.33E-01 (7.7%)	0.52	[-4.62, 1]
LSTM	α-shape	$[\gamma\ \theta\ q]$	2.55E-03 (0.2%)	2.76E-02 (2.1%)	0.65	[-0.34, 1]
LSTM	MPDF	$[V\ \gamma\ \theta\ q]$	2.84E-03 (0.3%)	8.04E-03 (0.8%)	0.96	[0, 1]



(a) (b) (c)

(d) (e) (f)

Figure 8: *Histories of the loss functions of the trained ANNs after every epoch computed using the training data set (orange) and the validation data set (blue). The ANN architectures are (a),(b),(c): FFNN (d),(e),(f): LSTM. The reference sets are (a),(d): α-shape in $V-\gamma-\theta$ (b),(e): α-shape in $\gamma-\theta-q$ (c),(f): MPDF in $V-\gamma-\theta-q$.*

the ANN trained with a MPDF and with an α-shape in Figures 10 and 9 respectively, which the thresholds of the colour labels are tabulated in Table 4. For Figure 10 where the reference set is a MPDF, the thresholds are selected as 2σ and 3σ. This can be interpreted as that the red points lie inside the darker contour in Figure 5, blue points inside the lighter contour but outside the darker contour and green points lie outside the lighter contour. In the case of the network with α-shape as its reference, the score smaller than 0 corresponds to an end state point predicted to lie outside the α-shape boundary in Figure 4, as described in Subsection 3.3.

In Figures 10 and 9, all control sequences are numerically simulated regardless of the prediction made by the ANN. However it is possible to also avoid simulations with control sequences that are likely to not yield a meaningful information about the reachable set boundary described by the predicted score, saving simulation time. For example it is possible to use the LSTM trained with the MPDF to only simulate control sequences predicted to have 'good' and 'excellent' scores out of 10000 randomly generated sequences. This results in rejecting 49.71% of the sequences which correspond to end states in red in Figure 10, yielding a more compact end state sample data set only consisting of blue and green points.

Although the general shapes of the point clouds in Figures 9 and 10 are similar, there exist discrepancies in the distribution of the scores labelled to the end states. It can be seen that while in Figure 10 the 'bad' samples indicated in red are more spread out surrounding the initial state, in Figure 9 the red points are more concentrated which is the most prominent in the $\theta - q$ projection shown in Figures 9(d) and 10(d).

The same observation about the scores derived using α-shapes from Figure 6 can be made from Figure 9 that there are regions in the $V - \gamma - \theta$ projection where the edges of the point clouds are labelled as 'bad', which can be found with larger values of V as can be seen in Figure 9(a). On the other hand with the ANN trained with a MPDF the scores labelled on the end states are more homogeneous, that there are not clear separations in the labelled scores. For example in the $\theta - q$ projection shown in Figure 10(d) only a small portion of the 'excellent' end states are on the edge of the point cloud. However the same set of 'excellent' end states in the $V - \gamma - \theta$ projection they indeed lie on the edge of the point cloud. This reflects how the MPDF is described in a four-dimensional space and not in a specific projection in the state space.

Table 5 presents the performances of the trained ANNs for pruning control sequence samples when integrated in a MCS-based reachability analysis for a quadrotor drone system. The left most column indicates which of the two types of reference reachable set representations is used to train the ANN as presented in Subsection 4.3. The second column describes different thresholds selected for the pruning system that divides control sequence samples to be rejected or to be simulated according to the score predicted by the ANN. The rejection rate presented in the next column describes the percentage of the sampled control sequences with worse predicted scores than the corresponding thresholds. The two right columns present the coverage representing the ratio of the rejected end states lying inside the convex hull formed with the accepted end state samples. The values computed with the true scores computed using Equation 10 instead of the predicted scores from the ANN are shown in side parentheses. Higher rejection rate leads to less simulations yielding a reduced computation time used for numerical integration, while the coverage can be used as an indication of how much of the rejected samples may have contributed towards estimating the boundary of the reachable set.

Comparing the rejection rates of the ANN against the true values in the parentheses, it can be observed that the rejection rates when using the ANN trained with an α-shape as a reference are lower for score thresholds of 0.03 and 0.01 than the scores computed using Equation 8, while the opposite is true with a stricter threshold of 0.005. This means that the ANN trained with an α-shape generally overestimates the score when the true score is above 0.01 and underestimates when the true score is below 0.005. On the other hand, the ANN trained with a MPDF tends to reject more samples regardless of the threshold suggesting that the predicted scores are in general higher hence worse than the true value.

The coverage of the convex hull in the three-dimensional projections using the ANNs are almost always lower than the end states selected with the true scores, which the differences become larger with stricter score thresholds. This means that selecting a score threshold to reject more samples lead to an increased risk of the ANN incorrectly classifying the score of a given control sequence. It can also be observed that the ANN trained with an α-shape shows the highest performance in the $\gamma - \theta - q$, where the coverage of the convex hull using the ANN is comparable to that with that using the true scores. On the other hand the performance is poor in the $V - \gamma - \theta$ projections, which the coverage with the most lenient threshold is 0.849, which means that out of the 22% of the rejected samples 15% of them lie outside the convex hull formed with the accepted samples. Assuming a reachable set of a dynamic system is not necessarily convex, this means that by saving 22% of simulation time at least 15% of the ignored samples were potentially valuable in estimating the boundary of the reachable set. As for the ANN trained with the MPDF, relatively high coverage can be observed with thresholds of 1.5σ and 2σ for both $V - \gamma - \theta$ and $\gamma - \theta - q$ projections.

From data presented in Table 5, it can be concluded

Table 4: *Classification thresholds for labelling the end state points as 'bad', 'good' and 'excellent' presented in Figures 10 and 9.*

Figure	'Bad' (red)	'Good' (blue)	'Excellent' (green)
Figure 9	$\hat{S}_\alpha > 0.01$	$0 \geq \hat{S}_\alpha > 0.01$	$\hat{S}_\alpha \leq 0$
Figure 10	$\hat{S}_{KDE} > e^{-2^2/2}$	$e^{-2^2/2} \geq \hat{S}_{KDE} > e^{-3^2/2}$	$\hat{S}_{KDE} \leq e^{-3^2/2}$

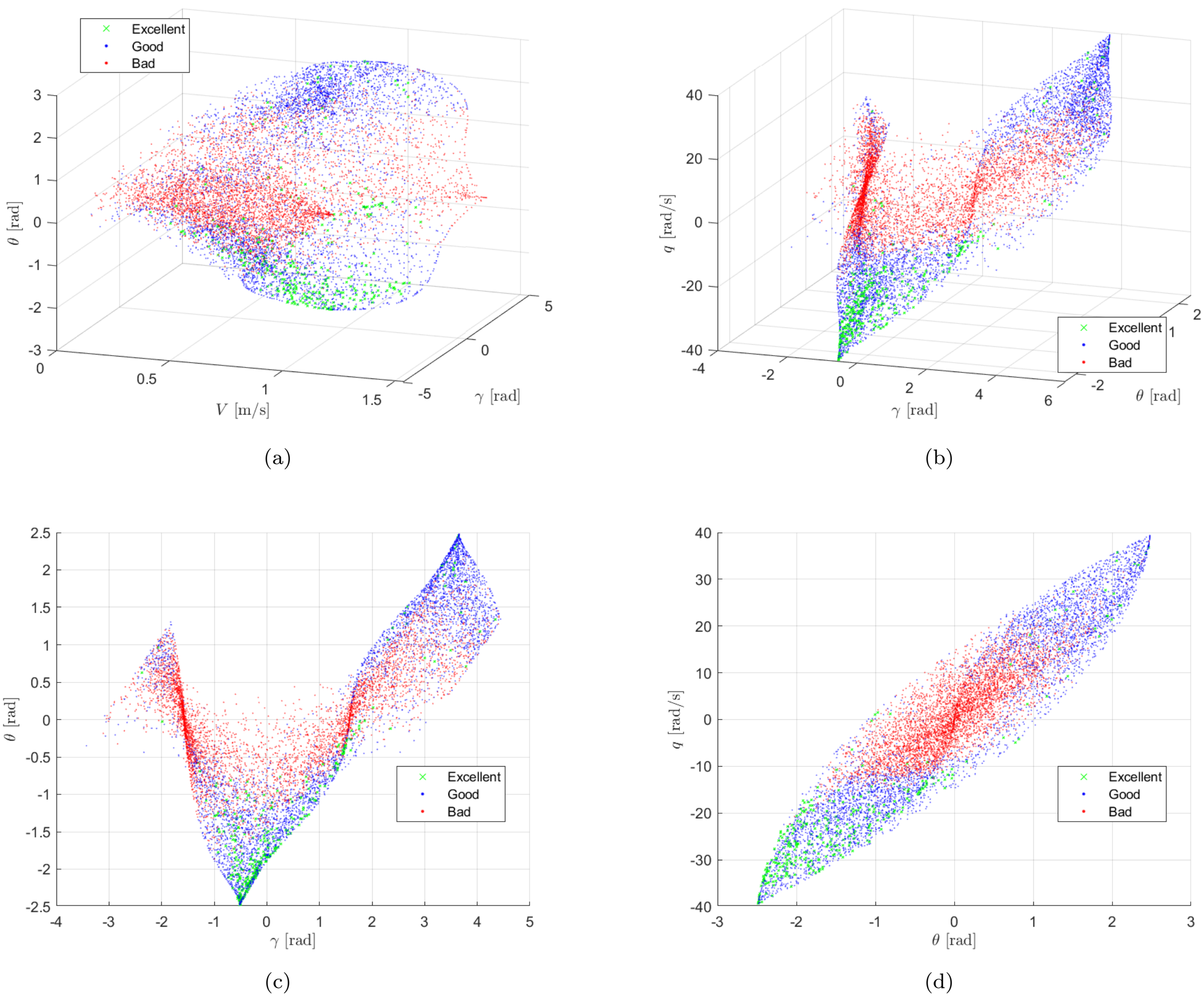


Figure 9: *10000 MCS end state samples of a longitudinal quadrotor drone with parameters in Table 1, labelled according to the score predicted by LSTM network trained with an α-shape in $\gamma - \theta - q$ as its reference reachable set shown in $V - \gamma - \theta$ and $\gamma - \theta - q$ slices. The points are labelled 'bad' in red, 'good' in blue and 'excellent' in green.*

Table 5: *Performance analysis result of the trained ANN in pruning control sequence samples to construct the reachable set boundary of a quadrotor drone system, compared against values calculated using the true scores shown in between parentheses.*

Reference type	Score threshold	Rejection rate	Coverage of convex hull	
			$V-\gamma-\theta$	$\gamma-\theta-q$
α-shape	0.03	22.44% (27.08%)	0.849 (0.904)	0.998 (1)
	0.01	49.52% (56.98%)	0.882 (0.937)	0.991 (1)
	0.005	73.05% (70.96%)	0.849 (0.946)	0.952 (1)
MPDF	1.5σ	22.78% (21.46%)	0.993 (1)	0.998 (0.999)
	2σ	49.71% (47.33%)	0.971 (0.974)	0.979 (0.971)
	2.5σ	75.04% (71.98%)	0.892 (0.928)	0.89 (0.923)

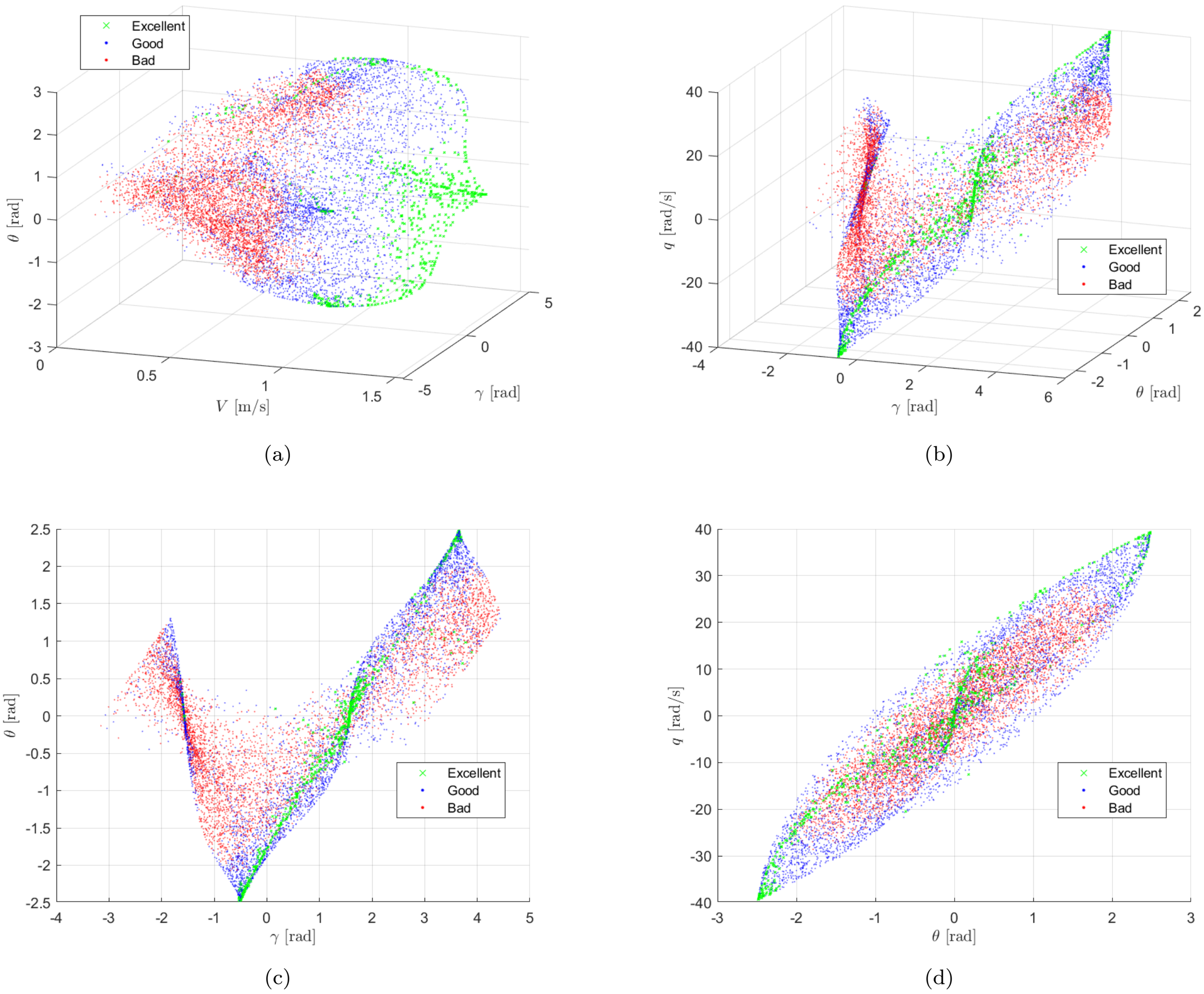


Figure 10: *10000 MCS end state samples of a longitudinal quadrotor drone with parameters in Table 1, labelled according to the score predicted by LSTM network trained with a MPDF as its reference reachable set shown in* $V - \gamma - \theta$ *and* $\gamma - \theta - q$ *slices. The points are labelled 'bad' in red, 'good' in blue and 'excellent' in green.*

that the ANN trained with a MPDF is generally able to predict the value of a control sequence better than the ANN trained with an α-shape for constructing a reachable set in the whole state space, not restricted to certain projections. In the following Subsection, the ANN trained with a MPDF is further validated for performing MCS-based reachability analysis with off-nominal systems with different model parameters compared to the nominal system used for training the ANN.

4.5 Validation with off-nominal systems

In order to assess the usefulness of the trained model in a real life application, the sensitivity of the performance of the ANN as a prediction tool to reduce the number of numerical simulations for reachability analysis is studied. This is performed by integrating it to a MCS of a quadrotor drone model with parameter changes compared to the values shown in Table 1. Two model parameters are selected for the sensitivity analysis, which are the time constant τ of the actuator dynamics and the vehicle mass m.

Table 6 tabulates the performances of the trained ANNs for pruning control sequence samples when integrated in a MCS using off-nominal dynamic systems. The types of data presented in Table 6 are identical to those in Table 5, except each row corresponds to a different dynamic model considered for the MCS. The nominal system is the quadrotor drone model to the parameters as presented in Table 1, where the remaining rows describe the differences compared to the nominal parameters.

When studying the general trend of the performance of the ANNs with varying score thresholds, it can be observed that the rejection rates with the same score thresholds using different simulated systems are almost identical. This is because the ANN only regards the control sequences sampled before simulating the dynamic system.

With the changes made to the actuator speed, by modifying the time constant of the first order system representing the actuator dynamics, changes in the performance can be observed. With a more lenient score threshold of 1.5σ, minimal changes can be observed in the coverage for both models with the slower and the faster actuators. However with stricter score thresholds, the model with the slower actuator yields end state samples that result in lower coverage than the nominal, where the differences are the largest with the score threshold of 2σ. The end states plots of simulations performed on the model with the slower actuators labelled according to the predicted scores are shown in Figure 11. It can be observed that the reachable set is smaller compared to Figure 10, which can be explained by how the actuators are slower hence the movement of the quadrotor is relatively more limited in a given time. However considering the distribution of the scores among the end state points, minimal changes can be observed compared to the nominal case in Figure 10.

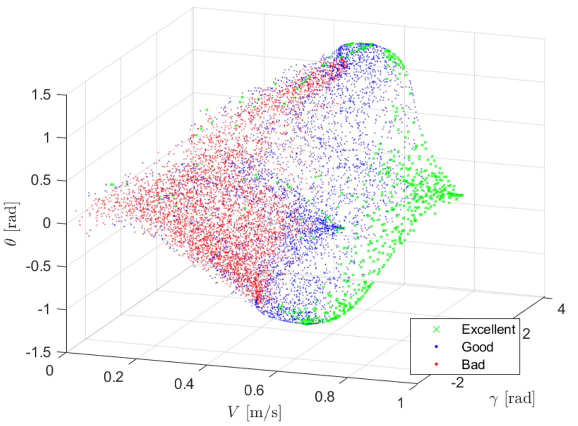
With the changes in the vehicle mass, limited changes in the coverage are observed with more lenient score thresholds of 1.5σ and 2σ. With a stricter score threshold of 2.5σ large reductions in the coverage can be observed with the model with lighter mass. On the other hand, the rejection rate with the model with larger mass is comparable with that with the nominal model. The plots of the end states of the model with 20% lighter mass labelled according to the predicted scores are shown in Figure 12. The lighter mass enables the system to achieve states further away from the initial hover condition, which results in a relatively larger reachable sets. However the general shapes and the distributions of the labelled scores are similar to those with the nominal model.

Although deterioration of the coverage is observed when the pruning system is integrated to off-nominal systems, it can be seen that with an appropriate choice of threshold for the predicted score the ANN can be used to prune trajectories as intended. This can be seen from the relatively high values of coverage shown in Table 4, which represents the ratio of the end states samples rejected by the ANN lying inside the convex hull formed with the accepted samples. For example with the score threshold of 2σ, the ANN is able to reduce the number of simulation using off-nominal systems by about 50% while at least 95% of the rejected samples lie within the convex hull formed by the remaining samples. The model parameter changes considered are 3 times slower actuators, 3 times faster actuators, 20% smaller vehicle mass and 20% larger mass than the nominal quadrotor drone model parameters shown in Table 1.

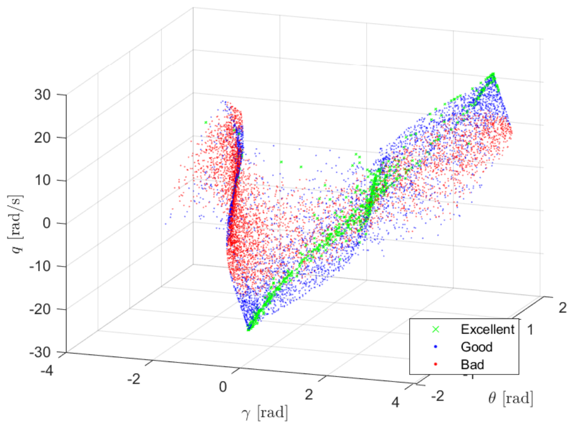
5 Conclusions

This paper presents a novel approach for integrating ANNs to reduce the number of simulations for flight envelope prediction of a quadrotor drone model using MCS by pruning randomly sampled control sequences. This is performed by training ANNs with supervised learning to map a randomly sampled control sequence to a scalar value that describes how close the corresponding trajectory leads towards the reference reachable set boundary, described as an α-shape or a MPDF. An LSTM network trained with a MPDF as its reference reachable set shows a high performance in identifying which sampled control sequences lead the dynamic model towards the boundary, which allows to reduce the number of numerical simulations.

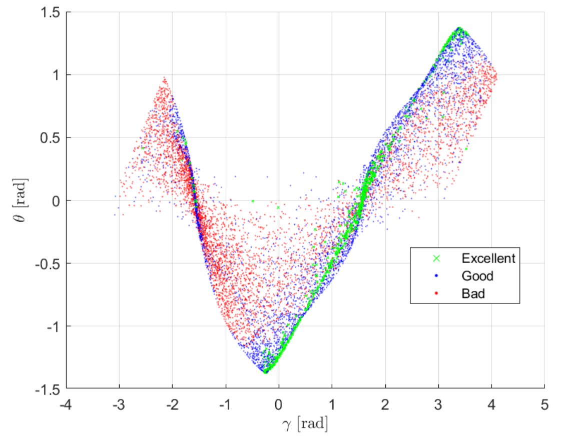
The ANN can be integrated in a MCS-based reachability analysis with different values of thresholds which



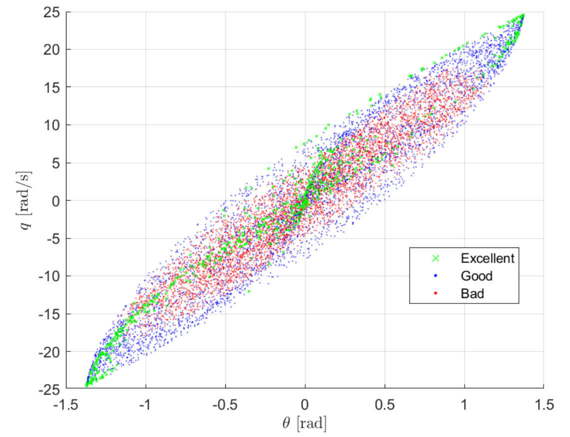
(a)



(b)



(c)



(d)

Figure 11: *10000 MCS end state samples of a longitudinal quadrotor drone with parameters in Table 1 except the time constant being* 1/10*, labelled according to the score predicted by LSTM network trained with a MPDF as its reference reachable set shown in* $V - \gamma - \theta$ *and* $\gamma - \theta - q$ *slices. The points are labelled 'bad' in red, 'good' in blue and 'excellent' in green.*

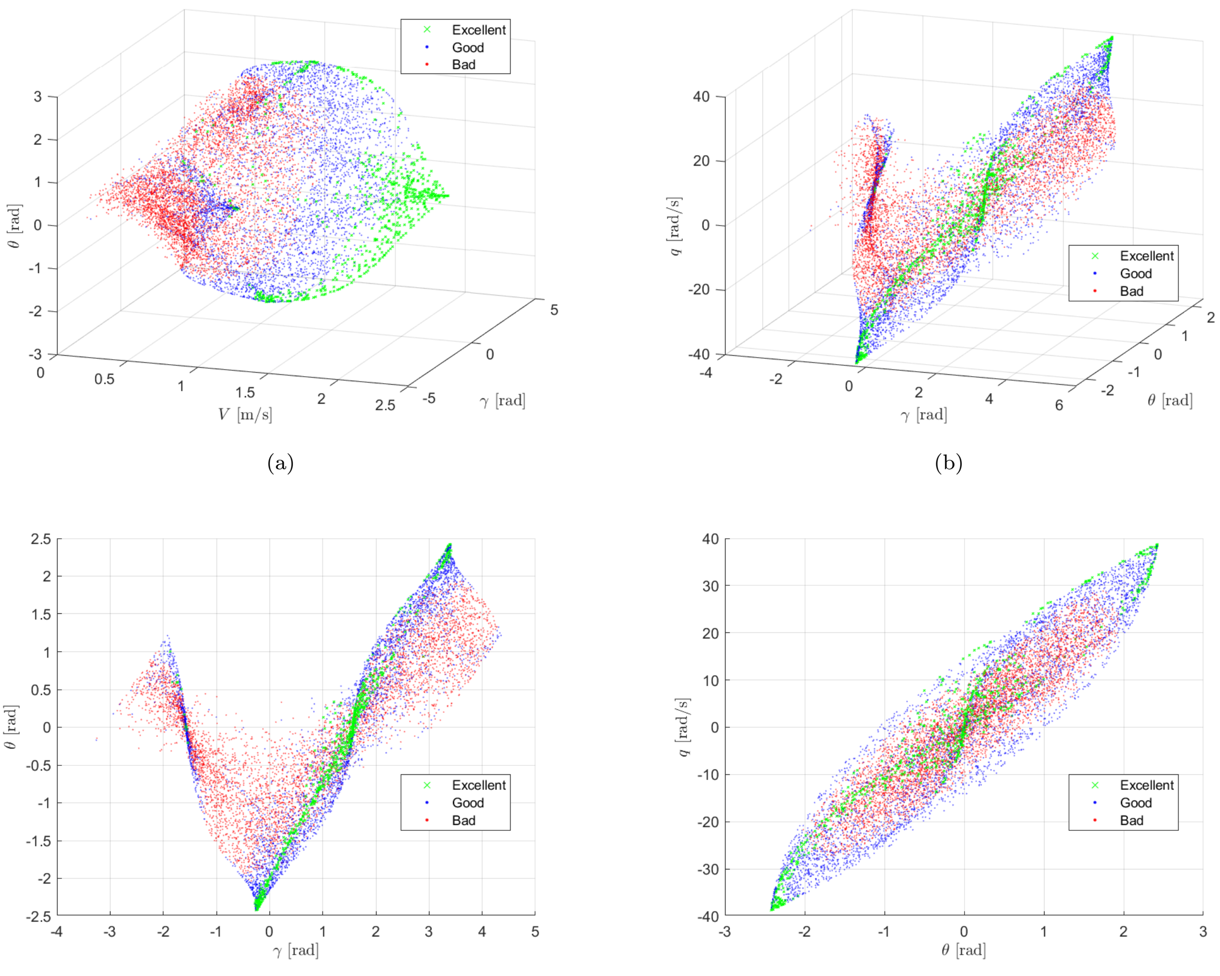


Figure 12: *10000 MCS end state samples of a longitudinal quadrotor drone with parameters in Table 1 except the vehicle mass being 20% lighter, labelled according to the score predicted by LSTM network trained with a MPDF as its reference reachable set shown in $V-\gamma-\theta$ and $\gamma-\theta-q$ slices. The points are labelled 'bad' in red, 'good' in blue and 'excellent' in green.*

Table 6: *Performance analysis result of the trained ANN in pruning control sequence samples to construct the reachable set boundary of a quadrotor drone with model parameter changes compared to the nominal model.*

Simulated system	Score threshold	Rejection rate	Coverage of convex hull	
			$V-\gamma-\theta$	$\gamma-\theta-q$
Nominal	1.5σ	22.78	0.993	0.998
Nominal	2σ	49.71	0.971	0.979
Nominal	2.5σ	75.04	0.892	0.89
3 times slower actuator	1.5σ	22.33	0.99	0.994
3 times slower actuator	2σ	49.42	0.947	0.948
3 times slower actuator	2.5σ	74.56	0.906	0.876
3 times faster actuator	1.5σ	22.79	0.995	0.997
3 times faster actuator	2σ	49.63	0.975	0.99
3 times faster actuator	2.5σ	74.13	0.929	0.886
20% smaller mass	1.5σ	23.09	0.979	0.993
20% smaller mass	2σ	49.92	0.966	0.981
20% smaller mass	2.5σ	74.33	0.635	0.777
20% larger mass	1.5σ	22.51	0.995	0.996
20% larger mass	2σ	50.03	0.98	0.98
20% larger mass	2.5σ	75.39	0.903	0.896

controls how much of the randomly sampled control sequences are rejected. Experiments show that with the threshold set to 2σ, the number of simulations to estimate the reachable set boundary can be reduced by approximately 50%, in which at most about 97% of the rejected end states lie within the convex hull formed with the accepted samples in three-dimensional projections of the state space. This suggests that at most 97% of the rejected samples do not contribute towards estimating the reachable set boundary. Further reduction in the numerical simulation can be achieved with a stricter threshold of 2.5σ resulting in a rejection rate of 75%, which comes at a cost of rejecting more of potentially valuable samples for boundary estimation.

The trained ANN shows a comparable performance when adapted to MCS on dynamic models with model parameter changes. These changes include three times slower and three times faster actuator dynamics, controlled by the time constant τ of the actuator dynamics modelled as first order systems, and 20% smaller and 20% larger vehicle mass. With all considered model parameter changes, the integrated ANN is able to reject about 50% of numerical simulations while at least 95% of the rejected end states lie within the convex hull formed with the accepted samples in two of the three-dimensional projections of the state space. This suggests that the ANN trained with this approach may also be effective in reducing the number of MCS samples to estimate the reachability set boundary of off-nominal dynamic systems with differences in the model parameters.

To draw conclusions about the effectiveness of this approach to reduce computation time for MCS reachability analysis, further research is recommended including an in-depth analysis of the computation cost comparison between a numerical simulation and an analysis of the LSTM network for a given sampled control sequence, as well as application of this method to more complex quadrotor drone dynamic models which models the dynamics of the system with higher accuracy. Further research for more optimal ANN architectures and training procedures is also recommended to further study the usefulness of the novel approach.

References

[1] E. R. van Oort, Q. P. Chu, and J. A. Mulder. Maneuver Envelope Determination through Reachability Analysis. In *Advances in Aerospace Guidance, Navigation and Control*, pages 91–102, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[2] Statistical Summary of Commercial Jet Airplane Accidents Worldwide Operations | 1959 – 2020. Technical report, Boeing, Seattle, Washington 98124-2207, USA, 2021.

[3] A Statistical Analysis of Commercial Aviation Accidents 1958-2019. Technical report, Airbus S.A.S., 31707 Blagnac Cedex, France, 2020.

[4] James Wilborn and John Foster. Defining Commercial Transport Loss-of-Control: A Quantitative Approach. In *AIAA Atmospheric Flight Mechanics Conference and Exhibit*, volume 1, pages 205–215. American Institute of Aeronautics and Astronautics, Reston, Virigina, 8 2004.

[5] W. Falkena, C. Borst, Q. P. Chu, and J. A. Mulder. Investigation of Practical Flight Envelope Protection Systems for Small Aircraft. *Journal of Guidance, Control, and Dynamics*, 34(4):976–988, 7 2011.

[6] Nikolas Tekles, June Chongvisal, Enric Xargay, Ronald Choe, Donald A. Talleur, Naira Hovakimyan, and Christine M. Belcastro. Design of a Flight Envelope Protection System for NASA's Transport Class Model. *Journal of Guidance, Control, and Dynamics*, 40(4):863–877, 4 2017.

[7] Liang Tang, Michael Roemer, Jianhua Ge, Agamemnon Crassidis, JVR Prasad, and Christine Belcastro. Methodologies for Adaptive Flight Envelope Estimation and Protection. In *AIAA Guidance, Navigation, and Control Conference*, Reston, Virigina, 8 2009. American Institute of Aeronautics and Astronautics.

[8] Shahab Kaynama, Ian M. Mitchell, Meeko Oishi, and Guy A. Dumont. Scalable Safety-Preserving Robust Control Synthesis for Continuous-Time Linear Systems. *IEEE Transactions on Automatic Control*, 60(11):3065–3070, 11 2015.

[9] M. G. Goman, A. V. Khramtsovsky, and E. N. Kolesnikov. Evaluation of Aircraft Performance and Maneuverability by Computation of Attainable Equilibrium Sets. *Journal of Guidance, Control, and Dynamics*, 31(2):329–339, 3 2008.

[10] John Lygeros. On reachability and minimum cost optimal control. *Automatica*, 40(6):917–927, 6 2004.

[11] Matthew Lichter, Alec Bateman, and Gary Balas. Flight Test Evaluation of a Run-Time Stability Margin Estimation Tool. In *AIAA Guidance, Navigation, and Control Conference*, Reston, Virigina, 8 2009. American Institute of Aeronautics and Astronautics.

[12] Mingzhou Yin, Q. P. Chu, Y. Zhang, Michael A. Niestroy, and C. C. de Visser. Probabilistic Flight Envelope Estimation with Application to Unstable Overactuated Aircraft. *Journal of Guidance, Control, and Dynamics*, 42(12):2650–2663, 12 2019.

[13] Sihao Sun and Coen C. de Visser. Quadrotor Safe Flight Envelope Prediction in the High-Speed Regime: A Monte-Carlo Approach. In *AIAA Scitech 2019 Forum*, Reston, Virginia, 1 2019. American Institute of Aeronautics and Astronautics.

[14] Ian M. Mitchell. Comparing Forward and Backward Reachability as Tools for Safety Analysis. In *Hybrid Systems: Computation and Control*, pages 428–443. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[15] Sankaran Mahadevan. Monte carlo simulation. In Thomas A. Cruse, editor, *Reliability-Based Mechanical Design*, chapter 4, pages 123–146. CRC Press, 1 edition, 1997.

[16] Jork Stapel, Coen C. de Visser, Erik-Jan Van Kampen, and Q Ping Chu. Efficient Methods for Flight Envelope Estimation through Reachability Analysis. In *AIAA Guidance, Navigation, and Control Conference*, Reston, Virginia, 1 2016. American Institute of Aeronautics and Astronautics.

[17] Thomas Lombaerts, Stefan Schuet, Kevin Wheeler, Diana M. Acosta, and John Kaneshige. Safe Maneuvering Envelope Estimation based on a Physical Approach. In *AIAA Guidance, Navigation, and Control (GNC) Conference*, Reston, Virginia, 8 2013. American Institute of Aeronautics and Astronautics.

[18] Newton H. Campbell, Michael J. Acheson, and Irene M. Gregory. Dynamic Vehicle Assessment for Intelligent Contingency Management of Urban Air Mobility Vehicles. In *AIAA Scitech 2021 Forum*, pages 1–15, Reston, Virginia, 1 2021. American Institute of Aeronautics and Astronautics.

[19] H. Edelsbrunner, D. Kirkpatrick, and R. Seidel. On the shape of a set of points in the plane. *IEEE Transactions on Information Theory*, 29(4):551–559, 7 1983.

[20] Herbert Edelsbrunner and Ernst P. Mücke. Three-dimensional alpha shapes. *ACM Transactions on Graphics*, 13(1):43–72, 1 1994.

[21] B.W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman & Hall, 1st edition, 1998.

[22] Sihao Sun, Coen C. de Visser, and Qiping Chu. Quadrotor Gray-Box Model Identification from High-Speed Flight Data. *Journal of Aircraft*, 56(2):645–661, 3 2019.

[23] Sihao Sun, Rudi Schilder, and Coen C. de Visser. Identification of Quadrotor Aerodynamic Model from High Speed Flight Data. In *2018 AIAA Atmospheric Flight Mechanics Conference*, volume 0, Reston, Virginia, 1 2018. American Institute of Aeronautics and Astronautics.

[24] Niklas Kochdumper and Matthias Althoff. Sparse Polynomial Zonotopes: A Novel Set Representation for Reachability Analysis. *IEEE Transactions on Automatic Control*, 66(9):4043–4058, 9 2021.

[25] Felix A. Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to Forget: Continual Prediction with LSTM. *Neural Computation*, 12(10):2451–2471, 10 2000.

[26] M Athans and P L Falb. *Optimal Control*. McGraw Hill, 1966.

Part III

Discussions and Conclusions

8

Conclusions and recommendations

This thesis was aimed to contribute towards enhancing the safety of multi-rotor drone operations by improving existing methods of flight envelope prediction using MCS by integrating ANN. The research objective and the corresponding research questions to be answered were established and discussed in detail in Chapter 2. The results presented in the scientific paper in Part II were made with the aim to answer these research questions, in which the conclusions and the list of recommendations are summarised and discussed in this chapter.

8.1. Conclusions

First, the research questions presented in Chapter 2 are revisited and assessed to what extent they have been answered. The first question was regarding the ways to represent the theoretical reachable set where there is no analytical solution for. Two methods of representing the reachable sets were considered using MCS end state samples, which are the outer contour represented as α-shapes and MPDF estimated using KDE. Both methods had their own advantages and disadvantages, for example α-shape being able to construct reachable set boundaries with relatively less samples, but were restricted in two-dimensional and three-dimensional spaces. On the other hand MPDF is able to quantitatively describe how safe or unsafe a given state is, while also suffering from the curse of dimensionality due to it being defined on a grid. Both these representations of the reachable sets were able to describe the reachable sets with comparable accuracy as the level set method, a well established reachability analysis method, which results are shown in Appendix B. Although direct comparison to the level set method is not possible with the six-dimensional longitudinal quadrotor drone system including actuator dynamics, the MCS-based approach for reachability analysis is most likely also valid for higher dimensional spaces.

The second question was regarding the measures that could be used to distinguish trajectories that reach further towards the boundary of the reachable set. In this research the ‘score’, defined using either of the reachable set representation methods, was employed as this measure. With α-shape references the trajectories were labelled as a function of the euclidean distances from the end state to the reference boundary and to the initial state, rewarding for reaching further towards the boundary and penalising for staying close to the initial state. With the MPDF reachable set representations, the degree of membership was directly used as the score. Both scores showed the desired characteristic that a trajectory with a better score tend to be positioned further into the state space from the initial state.

The third question was considering the type of machine learning technique suitable for predicting the relative value of a sampled control sequence in estimating the reachable set boundary. Through a literature review, ANN was deemed to have a high potential in achieving this where two variations were considered in this research, FFNN and networks containing LSTM cells. Through experiments it was shown that the trained ANNs were able to accurately predict the value of the sampled control sequence, which were defined as the derived score. While LSTM is slower than FFNN, it presented higher accuracy due to its recurrent nature. Furthermore, LSTM has added benefit that it can be used with input sequences of various lengths, while FFNN is only compatible with input sequences with the same lengths as they need to be flattened first. Therefore there exists a trade-off between implementing LSTM for higher accuracy against implementing FFNN which is less accurate and less flexible but has both faster training and computation times.

The fourth question is regarding the overall computational complexity and the computational time saved through integrating the ANN to prune input samples for MCS-based reachability analysis. This highly depends on the implementations of the numerical integration and the ANN computation, as well as the level of complexity of the considered dynamic model. For example implementation of ANN to reachability analysis of a simple system like the double integrator system does not benefit in terms of computational time, as the numerical simulations are less computationally expensive which can be performed with larger time steps. However the inputs of the ANN are the commanded control sequence which its sampling rate can be determined arbitrarily, while the time step for a numerical integration of a dynamic system is bounded to ensure stability. Therefore it can be argued that for more complex systems the implementation of ANN may result in an increased efficiency of MCS-sampling, as it was shown that the ANN is able to accurately prune control sequences using a simple longitudinal quadrotor drone model. In order to fully answer this question, an in-depth analysis of the costs of computation actions in question is essential as well as a study on how these change with an increase in complexity, for example the number of dimension, of the dynamic system.

The research questions discussed were derived from the main research question, which reflects the main motivation of this research. The research question is restated below:

Main research question

> *"Is it possible to reduce the number of required sample trajectories to predict flight envelopes using the MCS approach by integrating machine learning techniques?"*

It was shown that it is indeed possible to train an ANN to predict whether or not for a given control sequence the corresponding end state reaches near the reachable set boundary, which can be used to reduce the number of numerical simulations to estimate the reachable set boundary. It was also shown that the trained ANN is still effective in pruning control sequences for off-nominal dynamic systems, with changes to the system parameters compared to the nominal system used for training the ANN. Furthermore an argument was made that this approach for reachability analysis is also able to reduce the computational time in collecting end state samples to estimate reachable set boundaries.

8.2. Discussions and recommendations

A list of recommendations is presented that are aimed to further realise the research objective presented in Chapter 1. These are derived through analysing the limitations of the proposed approach and by considering the necessary steps to make the novel approach more useful in real life applications.

Benefits and limitations of the proposed approach

The core idea of the proposed approach is that the each numerical simulation performed for a MCS-based reachability analyses varies in its usefulness when constructing the boundary of the reachable set, described by the scores labelled on each of the simulation samples. Which the proposed approach uses this idea to circumvent performing numerical simulations that are likely to be of less importance when estimating the reachable set boundary, by associating each sampled control sequence to the scores using ANNs.

The main added benefit of this approach over purely probabilistic approach, for example by Yin et al. [14] and Sun and de Visser [15], is that information about the usefulness of the sample can be derived from the control sequence before performing computationally expensive numerical simulations. This information can be used as a reference to ignore a large portion of simulations as presented in this research, or the scores themselves may also be used to reconstruct the reachable set estimation. With the reachable set described as a MPDF, the resultant end state and the predicted score (the membership function) of the control sequence together form an input-output pair of the MPDF itself. This may for example be used to adaptively estimate the probabilistic reachable set represented as a MPDF.

Another benefit is by training an ANN using the MCS results of a specific dynamic system, reduction in the number of simulations can be achieved which is specific to that system opposing the purely probabilistic approach which reduces the number of simulations in a more general way. This is motivated by how a specific control sequence can have largely different scores depending on the dynamic system, as observed between the double integrator problem and the inverted pendulum problem shown in Appendix B. Furthermore the ANNs have high flexibility in its input and output formats, compatible with for example a non-bang-bang control which may be more optimal than bang-bang controls for nonlinear systems.

On the other hand, the proposed approach has several limitations. First, the achieved reduction in computation time and complexity from the integration of an ANN may not be as significant since ANNs can also be relatively computationally expensive. Simply storing sequences with relatively good scores may be sufficient to reduce the number of simulations. Further development of the proposed approach is recommended to benefit from the prediction power of the ANN, for example for an adaptive reachability analysis where the dynamics of the system changes over time.

Another limitation is that this approach requires a step to generate the reachable set before labelling end state samples. This requires an accurate knowledge of the dynamic model as well as time and resources to collect simulation samples and to train the ANN. The resultant ANN may be used in the future reachability analyses on the same or similar dynamic systems, however this approach cannot be directly applied to an unknown system.

Finally, the expected outcome of employing the proposed framework is a more compact end state samples that lie near the reachable set of the dynamic system. However these end state samples need to be synthesised as a continuous set to formulate the flight envelope which can be used for flight envelope protection, which is also a challenging process. A MPDF derived from this compact data set cannot be used as an indication of the safety of the system, as the samples in the regions likely to be reached are eliminated making them relatively 'unsafe'. Outer contours may be constructed with for example α-shapes, however there are risks of under- or overestimating the reachable set which are difficult to identify when the theoretical reachable set is unknown.

The framework of MCS-based reachability analysis also has unsolved problems. One significant element is the choice of the time horizon. With a time horizon that is too short the corresponding reachable set may only cover a very small range around the a-priori known safe set. On the other hand with a time horizon that is too long the reachable set becomes so large that no meaningful knowledge about the safety of the system can be derived, especially for highly manoeuvrable systems like the quadrotor drones. Development of more systematic methods of choosing an appropriate time horizon to determine the safe flight envelope of a given system is of high importance, yet to this date not a large body of literature can be found.

Another crucial aspect to consider is the validity of treating flight envelopes as probabilistic sets. Employing the MCS is performed by randomly generating a control sequence, which is not how an actual control sequence is commended to the systems. This means that there exist control sequences that are technically possible but are almost never performed during flight. In order to derive an accurate probabilistic safety of a given state during flight, a more sophisticated sampling strategy may need to be developed that resembles the probability distribution of control sequences performed during flight either by a flight controller or a pilot.

Optimising the current approach

There are different aspects in the current approach which its efficiency and accuracy in reducing simulation samples in MCS can be improved, but could not be fully addressed due to the time constraint. One of them is the optimisation of ANN architecture design and training processes. In this research the main design choice considered in the architecture design is whether or not the ANN has a recurrent characteristic, while comparatively less attention was paid on the effects of the choices of hyper-parameters. These include but are not limited to the number of neurons, number of hidden layers and the choice of activation functions. Research on the effects of design choices of the training process such as the loss function, number of epochs and different regularisation methods are also recommended.

Another aspect that may be optimised is the definition of the scores labelled on each end state. This process has a great amount of freedom from the choice of the representation method of the reachable set to choices on which elements of the end state to reward or penalise. It may also possible to define a score metric that considers multiple representations of the reachable sets. For example in a four dimensional state space, the reachable set may be constructed from MCS samples as outer contours in two three-dimensional projections, where the score of an end state can be derived as the weighted average of the distances of the end state to the boundaries in these two projections.

Further analysis of the usefulness of the proposed approach

The usefulness of the proposed approach is presented in the scientific paper, which may be verified further by applying it to more sophisticated systems. The dynamic model of the quadrotor considered in this research is simplistic, while more complex models with for example look up tables are available with higher accuracy. Application of the proposed approach to such systems is recommended to study the effectiveness of the proposed approach.

The effect of the stochasticity on the performance of the proposed approach may also be of high value for applications in real life situations. Introduction of stochasticity in this approach is relatively straightforward, for example adding noise to the states during numerical simulations and to add noise to model parameters in every simulation.

It is expected that with more complex dynamic systems the training of ANN becomes more challenging to maintain certain level of accuracy. This may require more sophisticated ANN architectures, increased size of training data sets and longer training times. The extent of these required changes compared to the increase in the complexity of the model is recommended to be studied to further analyse the usefulness of the novel approach.

Extension of the approach

There are a number of ways which the proposed approach can be extended upon to further realise the research objective. One way is to develop a training scheme that can be performed on an already trained ANN. This can be used for example to speed up the training process of the ANN using dynamic systems with similar characteristics as the nominal system. Another extension to the ANN training process is to also consider model parameters of the dynamic system as an input for the ANN. This means that the ANN predicts the value of a given random control sequence sample for estimating the reachable set boundary of the dynamic system with specific model parameters. This ANN can be used for different dynamic systems, for example different quadrotor drone models or different failure modes of the system.

One way to effectively use the benefits brought by the novel approach is to develop a way to adaptively modify the reachable set representation as new simulations are made. If the basic knowledge of the reachable set, such as its general shape, an accurate representation of the reachable set may be estimated quickly by combining the a-priori known information and the compact set of end states collected through the proposed approach. In this process the predicted score of the system may also be used to reevaluate the reachable set. For example with an ANN predicting the score using a MPDF, which is the membership function of the MPDF, the new end state sample and its score together form an input-output pair of the MPDF itself.

Part IV

Appendices

A

Documentations

The MCS-based reachability analysis and the ANN training were implemented in Python and MATLAB. In this chapter the implementations are briefly described.

A.1. Monte-Carlo simulation implementations

`save_random_sequence.py`

Generates and saves random sequences consisting of 0 and 1, corresponding to minimum and maximum commanded input to be used for MCS. Parameter p_c can be changed to control the extremeness of generated control sequences.

`QD_hov.py`

The main code for numerically simulating quadrotor drone models based on the simulation environment developed by Jasper van Beers. Within the code, the dynamic model, equations of motion and actuator dynamics are defined. This code takes in randomly generated sequences from `save_random_sequence.py`, performs numerical integration with each of the sequences and saves the final states reached after each simulation. Similar codes named `DI.py` and `IP.py` were used to simulate double integrator and inverted pendulum systems respectively.

A.2. Set construction and score assignment implementation

`genAlphaShape.m`

Using the end states data, α-shapes are created in three-dimensional projections of the state space as `alphaShape` objects using the `alphaShape` function in MATLAB. This code adjusts the parameter α as a minimum value where a specified ratio of the validation end states lie within the resultant α-shape.

`save_alphashape.m`

The `genAlphaShape.m` is called to create and save α-shapes. The α-shapes are saved as `alphaShape` objects.

`save_KDE.m`

Using the end state data, MPDF are created on a gridded four-dimensional space using the `mvksdensity` function in MATLAB. The MPDF is saved as a multi-dimensional grid and a multi-dimensional array of the MPDF values of the corresponding grid points.

save_alpha_score.m
This code regards the end state of a trajectory and assigns the score using the reference α-shape to the corresponding control sequence, using the equation shown in the scientific paper.

save_KDE_score.m
This code regards the end state of a trajectory and assigns the score using the reference MPDF to the corresponding control sequence, using the equation shown in the scientific paper.

LS_Two_dcart.m
The code to compute the reachable set as a level set of the double integrator (2D cart), using the toolbox developed by Prashant Solanki based on the toolbox by Mitchell. Similarly, `LS_IP.m` generates the level set of the inverted pendulum system.

A.3. ANN training and trajectory pruning

train_score.py
The main code to perform ANN training, where the ANNs are constructed and trained using Keras [63]. The ANN model is constructed in the class `NeuralNetClass.py` and the training data is loaded and pre-processed with the class `TrainingDataClass.py`. Each trained ANN is saved as a Keras `Model` class together with model description, training data description and a log of loss function after each epoch.

analyse_performance.py
The code to analyse the performance of the trained ANN. This code analyses the mean and the standard deviation of the loss function (squared error) and the cross correlation using the validation data set. This code also creates plots of the loss function histories.

NN_filter_QDhov.py
The code that integrates the trained ANN to MCS-based reachability analysis. A random sequence is generated then fed to the imported trained ANN, which then is numerically simulated using `QD_hov.py`. The end state samples are saved together with the predicted score. By applying a threshold the code ignores randomly generated sequences with worse predicted score than the threshold.

B
Additional results

B.1. Monte-Carlo simulation

Figure B.1 shows how the randomly sampled control sequence is converted to the actual control of the quadrotor drone model with the effect of the actuator dynamics. It can be seen that for the forward simulation the actual control starts from the trim speed shown with blue dotted lines. On the other hand, with the reverse simulation the control at $t = 0$ is not necessarily at trim as described in the main paper. However this ensures that the reverse trajectory can be traced exactly forward in time with the actuator dynamic modelled with the same first order system.

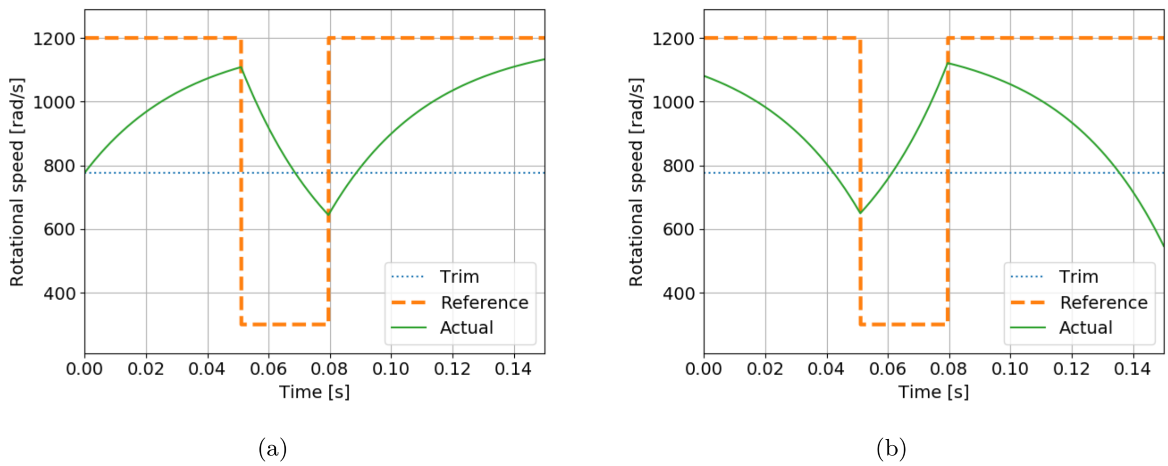
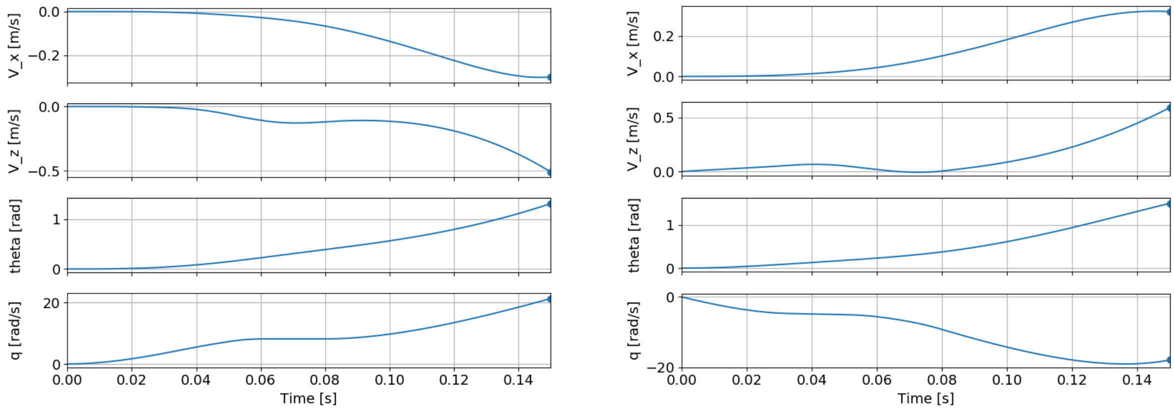


Figure B.1: Actual control sequence of the quadrotor drone with actuator dynamics for (a) forward and (b) reverse simulations, subject to the same reference control signal.

Figure B.2 shows the history of the states of a quadrotor drone simulation performed with a randomly sampled control sequence. The states shown are those of the EOM of the quadrotor drone $V_x - V_z - \theta - q$, before being converted to the $V - \gamma - \theta - q$ space. The states reached at the end are collected to form a point cloud, in which a continuous set can be derived which serves as an estimation of the reachable set.

The numerical integration is performed using the 4th order Runge-Kutta scheme, where the reference sequences have lengths of 100 where integration step is set as 15 times that resulting in 1500 integration steps. This number is chosen by comparing the forward simulations of 0.15

seconds with randomly sampled sequences and then performing reverse simulations from their end states with the identical control sequence flipped in time, in which the reverse simulations should end at the initial state. With larger integration steps the end state of the reverse simulation yielded large deviations from the initial state of up to 10% of the largest deviation observed in the trajectory in either of the four states. Same experiment was performed for both the double integrator and the inverted pendulum problems, however the same number of integration steps as the length of the control sequence yielded very small error between the forward and the reverse trajectories.



(a) (b)

Figure B.2: State trajectory of a quadrotor drone model subject to a randomly sampled control sequence from (a) forward and (b) reverse simulations.

B.2. Reference set construction

Figure B.3 shows a set of plots resulting from the MCS-based reachability analysis performed on the double integrator system, presented in Chapter 7. The sampled end states and the resultant reachable set estimations are verified and validated using the level set method and the analytical solution. The reachable set of the double integrator system estimated using the level set method is shown in Figure B.3(c), which presents the initial level set and propagated level sets after 0.5, 1.0 and 1.5 seconds. Since the level set method is performed on a gridded space, the initial set is not a point set but a very small continuous set formed by points on the gridded space. It can be observed from Figure B.3(c) that the reachable set changes its shape with increasing length of time horizon, not simply expanding in size about the origin like in Figure B.3(b). This suggests that the different levels of contour obtained with the MPDF cannot be used to interpolate nor extrapolate the reachable set with a different time horizon, at least with the sampling strategy of the MCS used in this research.

Figure B.3(d) shows the MCS end state samples together with the contour of the analytical solution at $\phi = 1$. The contour fully encapsulates the end state samples, meaning that each numerical simulation performed on the dynamic system yields an end state sample that is strictly conservative. Furthermore, It can be observed that the contour is almost identical to the convex hull shown in Figure B.3(a). For the double integrator system the MCS is able to produce a reachable set boundary estimate with a comparable or even higher accuracy than the level set method with the considered grid size.

Figures B.4, B.5 and B.6 show the MCS-based reachability analysis results of the inverted pendulum system using α-shape with MCS samples, MPDF with MCS samples and the level

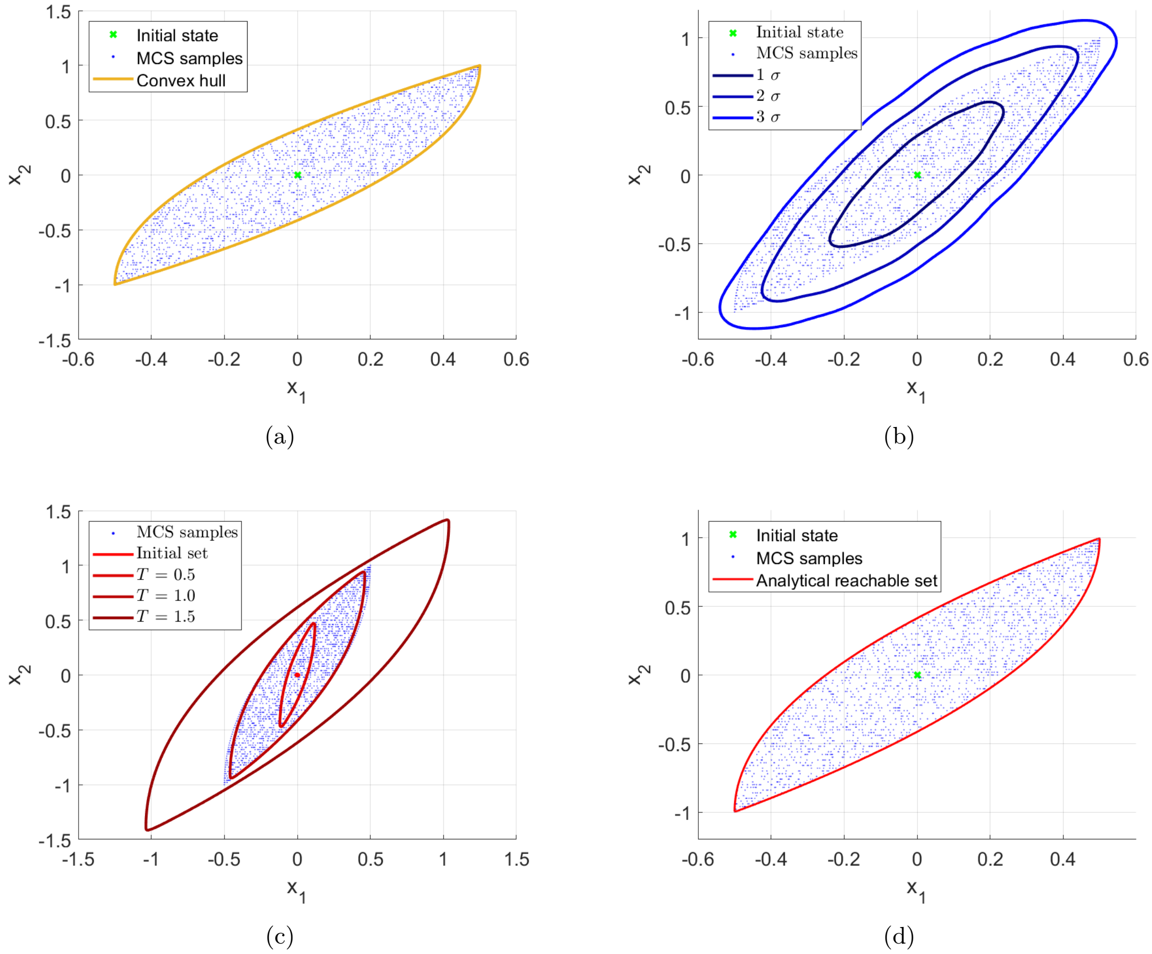


Figure B.3: Representations of the reachable set of double integrator problem of time horizon of 1 second using: (a) convex hull of MCS samples (b) MPDF isocontours of MCS samples (c) the level set method (d) analytical solution

set method respectively. The initial set is relatively larger compared to Figure B.3(c), while the MCS end states are sampled with simulations starting from the initial point at $[0,0,\pi,0]$. However it can be seen that the MCS end state point cloud forms a similar shape as the level set.

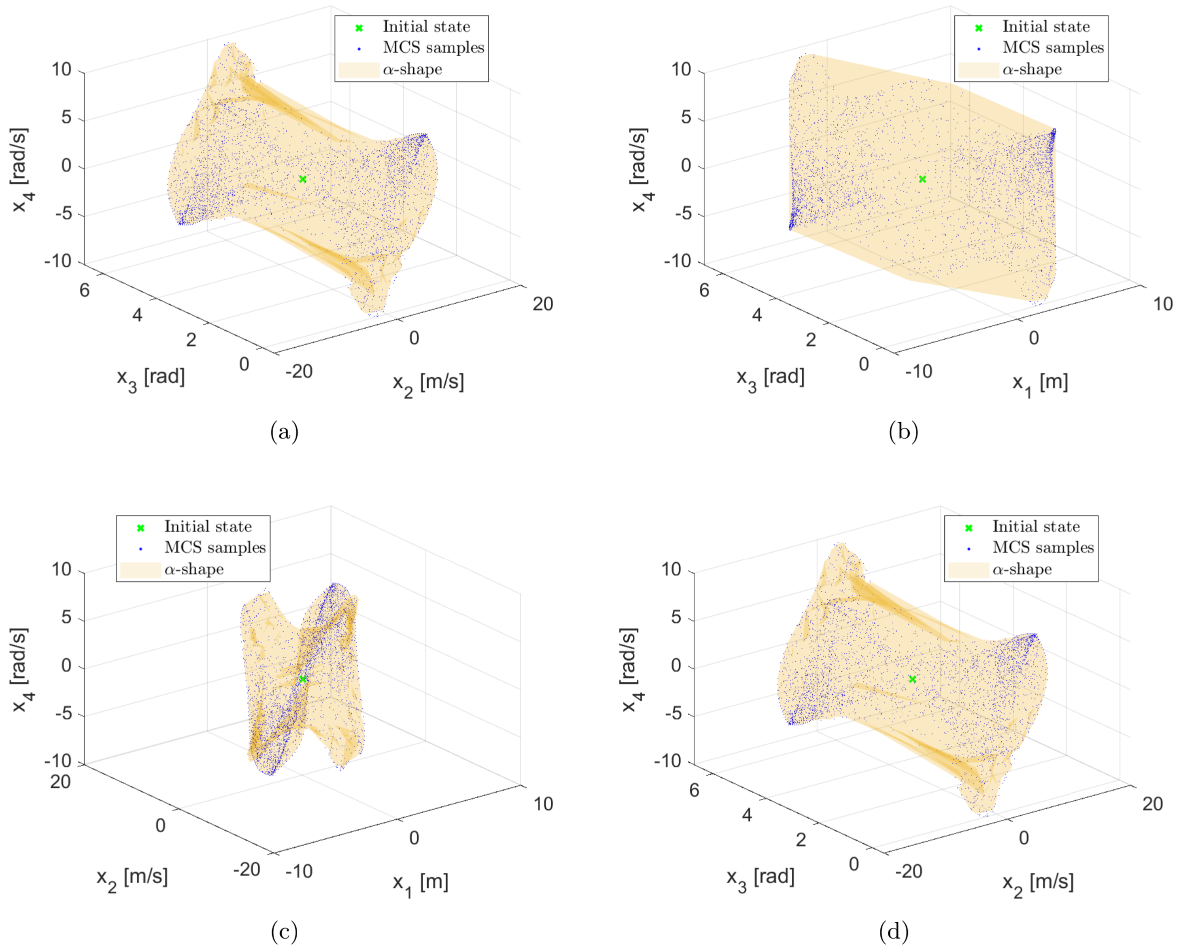


Figure B.4: Reachable set boundary estimated as an α-shape of MCS samples of the inverted pendulum system with time horizon of 1 second.

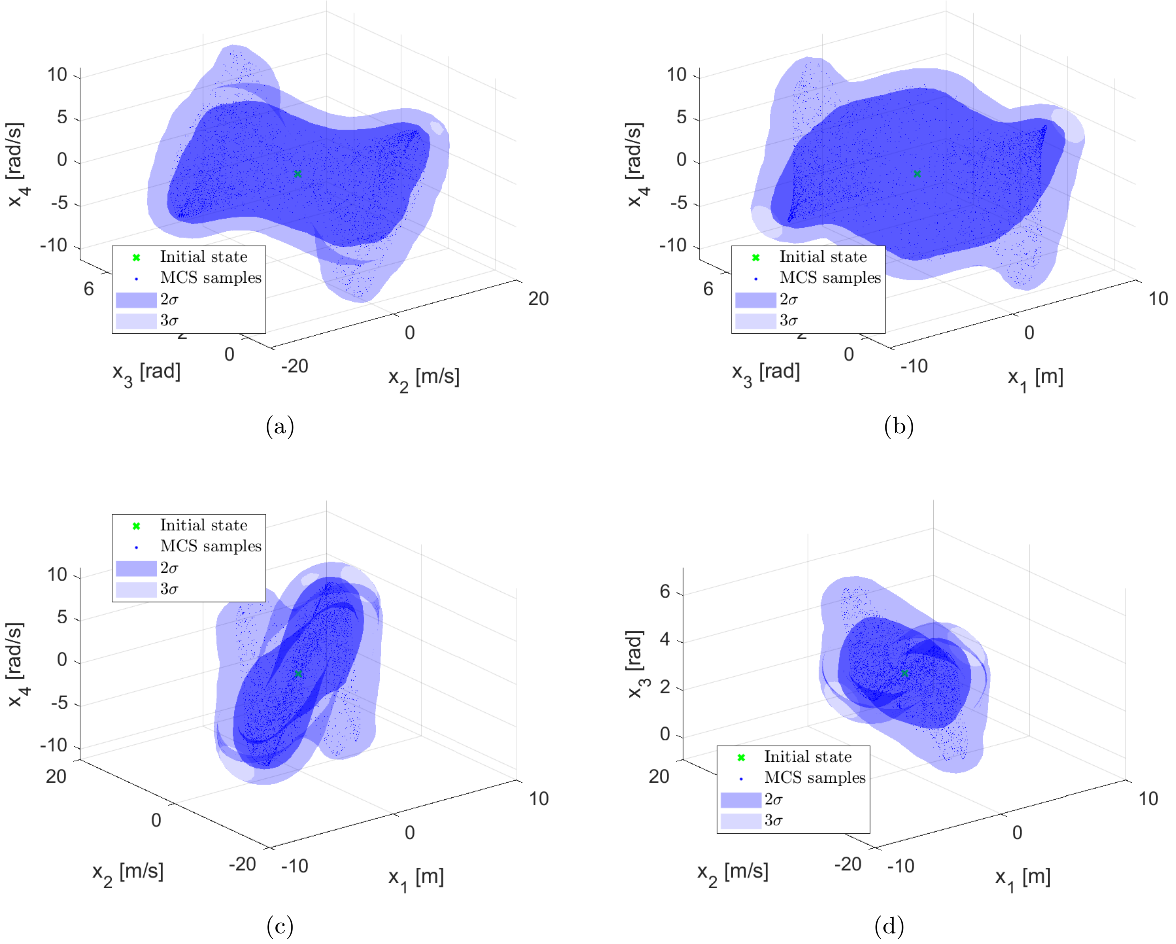
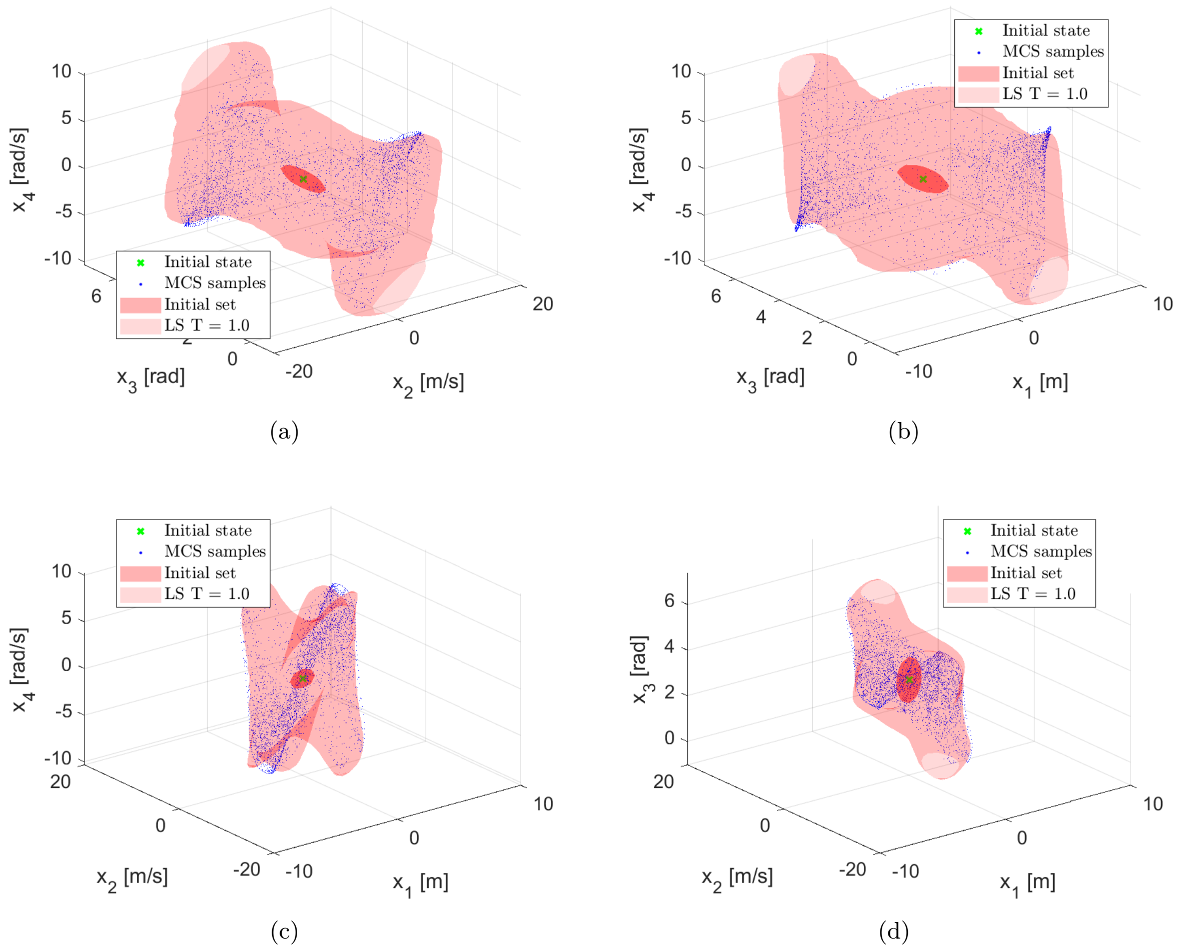


Figure B.5: Reachable set estimated MPDF of MCS samples of the inverted pendulum system with time horizon of 1 second, visualised as isocontours.



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

Figure B.6: Reachable set boundary estimated of MCS samples of the inverted pendulum system with time horizon of 1 second using the level set method.

B.3. Trajectory labelling

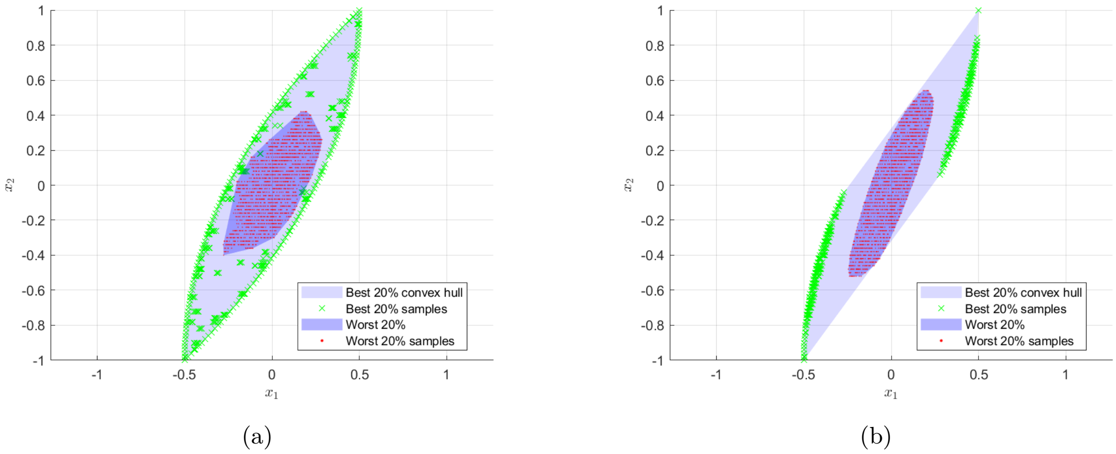


Figure B.7: Verification result of the scoring system applied on the double integrator system using: (a) α-shape reference set (b) MPDF reference set

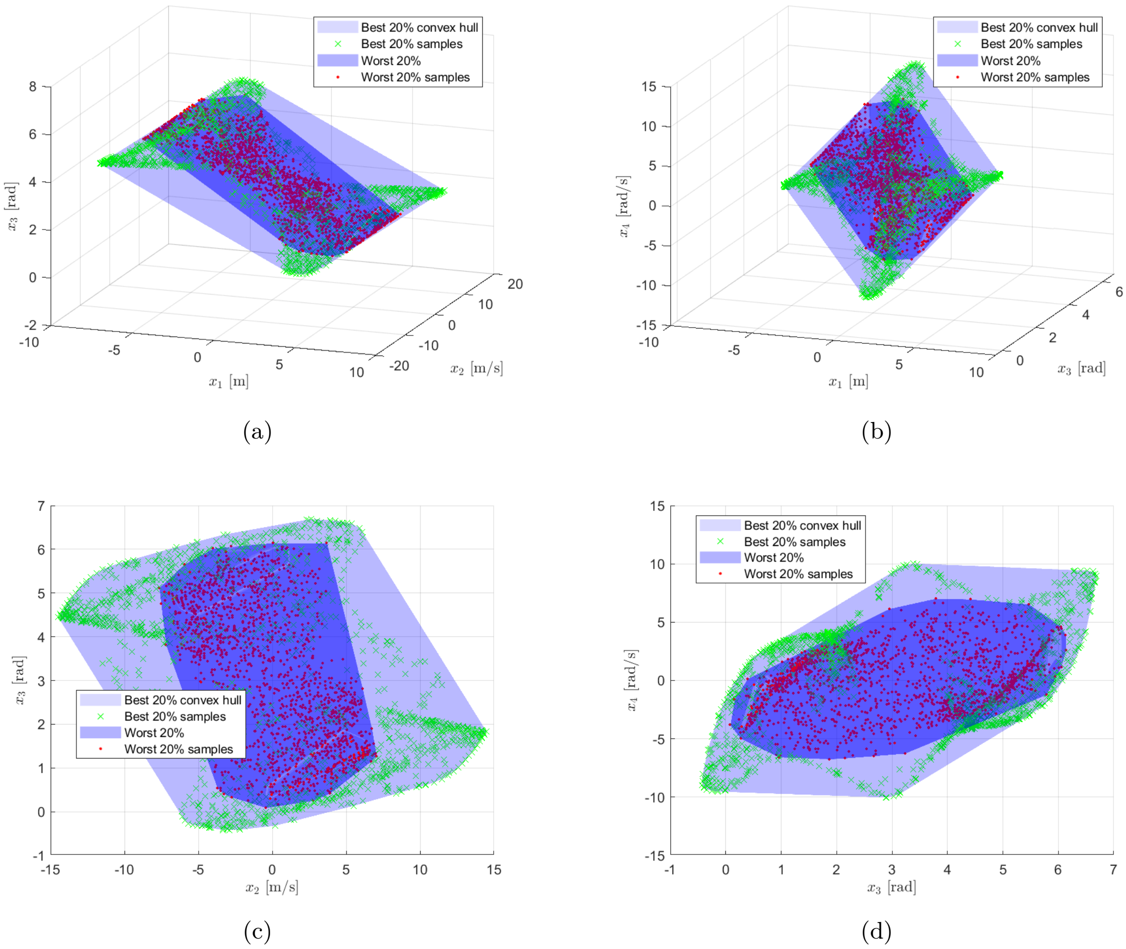


Figure B.8: Verification results of scores applied on the inverted pendulum system using α-shape reference set in three-dimensional projections.

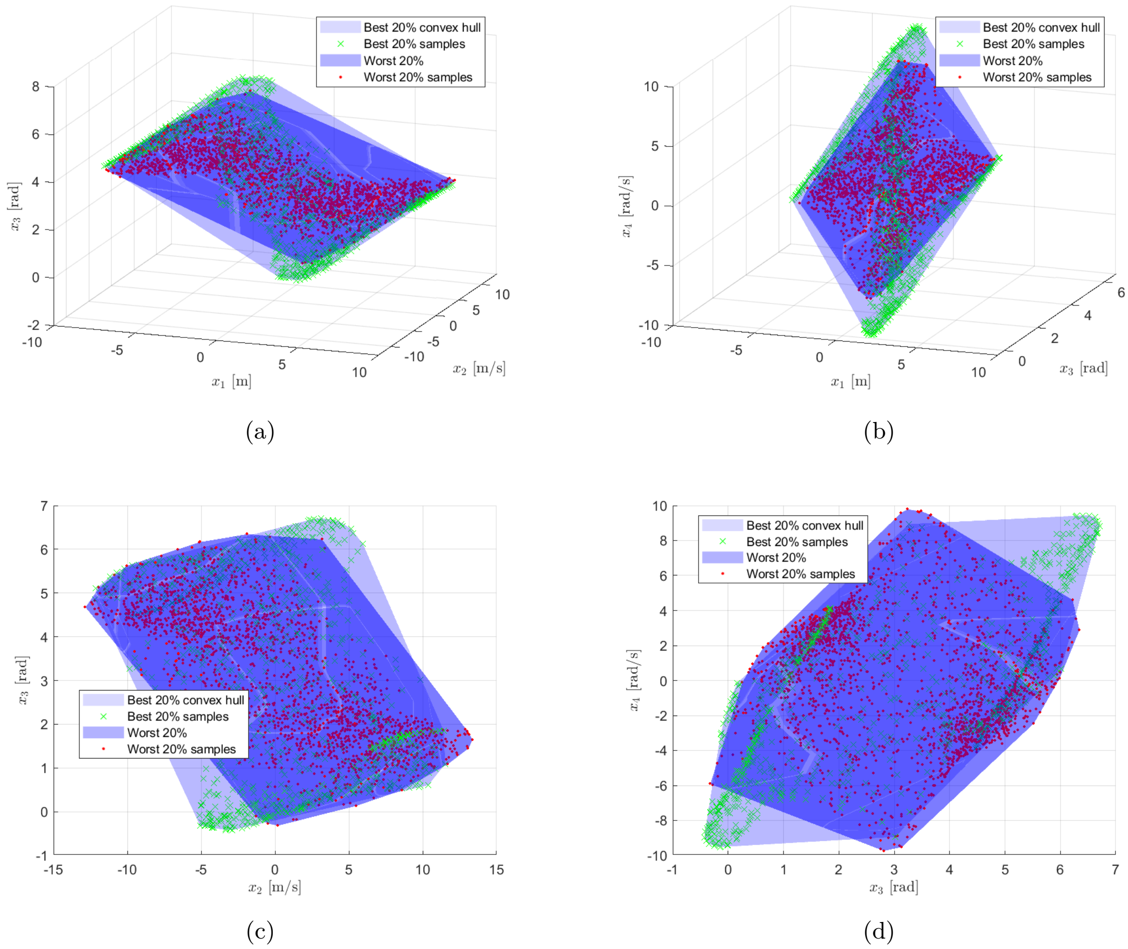


Figure B.9: Verification results of scores applied on the inverted pendulum system using the MPDF reference set shown in three-dimensional projections.

B.4. Artificial neural network training

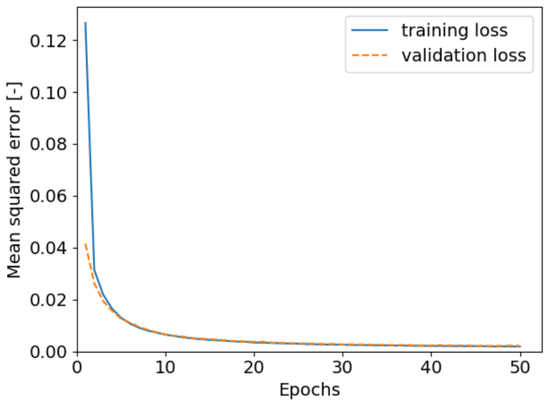
Tables B.1 and B.2 tabulate the performance results of ANNs trained for the double integrator and the inverted pendulum systems respectively. The ANNs are trained with the same methodology as the quadrotor drone, except with different numbers of epochs. The number of epochs for the ANNS using the double integrator and the inverted pendulum systems are 50 and 100, respectively. The histories of loss functions during training of the ANNs are shown in Figures B.10 and B.11. Highest correlation is observed with LSTMs trained with MPDF reference sets, as also observed using the quadrotor drone system.

Table B.1: Performance analysis result of the ANNs trained to map the commanded control sequence to the score of the corresponding end state of the double integrator system

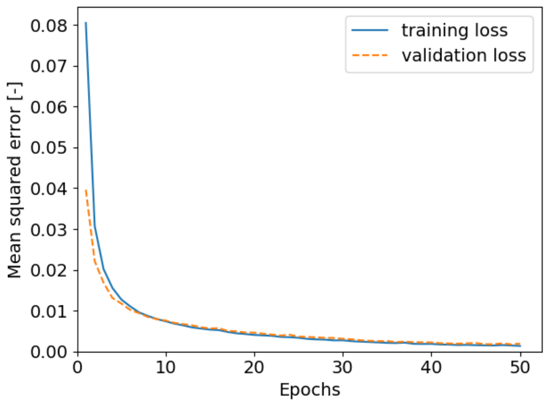
ANN type	Reference set type	Projected space	MSE (relative to score range)	std SE (relative to score range)	Correlation	Score range
FFNN	α-shape	$x_1 - x_2$	2.58E-03 (0.3%)	4.02E-01 (39.7%)	0.56	[-0.013,1]
FFNN	MPDF	$x_1 - x_2$	1.85E-03(0.2%)	5.33E-02(5.4%)	0.99	[0.011,1]
LSTM	α-shape	$x_1 - x_2$	1.82E-03(0.2%)	5.76E-01(56.9%)	0.71	[-0.013,1]
LSTM	MPDF	$x_1 - x_2$	6.15E-04(0.1%)	1.19E-02(1.2%)	1.00	[0.011,1]

Table B.2: Performance analysis result of the ANNs trained to map the commanded control sequence to the score of the corresponding end state of the inverted pendulum system

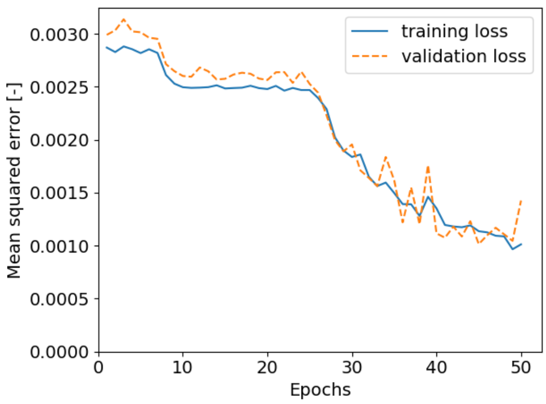
ANN type	Reference set type	Projected space	MSE (relative to score range)	std SE (relative to score range)	Correlation	Score range
FFNN	α-shape	$x_1 - x_2 - x_4$	5.98E-04 (0.06%)	2.83E-03 (0.27%)	0.44	[-0.05,1]
FFNN	α-shape	$x_2 - x_3 - x_4$	5.00E-04 (0.05%)	3.12E-03 (0.30%)	0.37	[-0.03,1]
FFNN	MPDF	$x_1 - x_2 - x_3 - x_4$	5.60E-03 (0.57%)	1.36E-02 (1.38%)	0.91	[0.01,1]
LSTM	α-shape	$x_1 - x_2 - x_4$	4.75E-04 (0.05%)	3.99E-03 (0.38%)	0.37	[-0.05,1]
LSTM	α-shape	$x_2 - x_3 - x_4$	3.53E-04 (0.03%)	3.19E-03 (0.31%)	0.47	[-0.03,1]
LSTM	MPDF	$x_1 - x_2 - x_3 - x_4$	2.89E-03(0.29%)	9.78E-03 (0.99%)	0.96	[0.01,1]



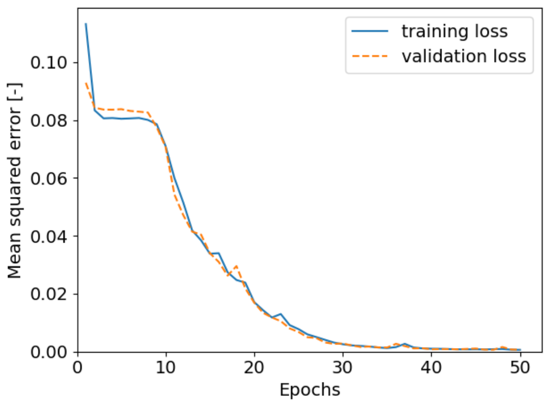
(a)



(b)

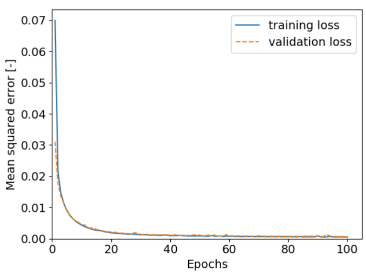


(c)

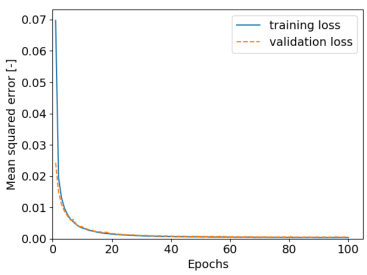


(d)

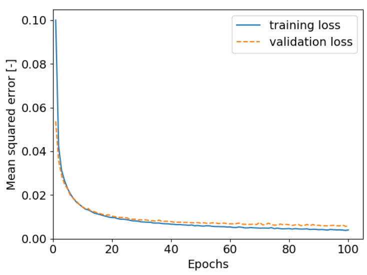
Figure B.10: Histories of the loss functions of the trained ANNs after every epoch computed using the training data set (orange) and the validation data set (blue). The ANN architectures are (a),(b): FFNN (c),(d): LSTM. The reference sets are (a),(c): α-shape in $x_1 - x_2$ (b),(d): MPDF in $x_1 - x_2$.



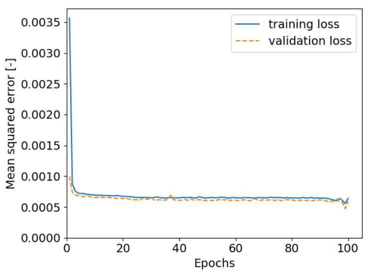
(a)



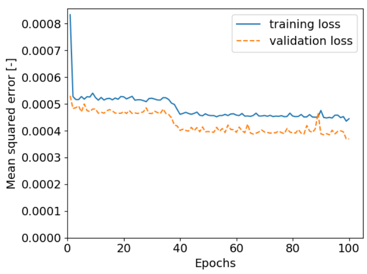
(b)



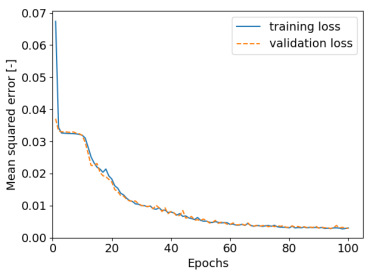
(c)



(d)



(e)



(f)

Figure B.11: Histories of the loss functions of the trained ANNs after every epoch computed using the training data set (orange) and the validation data set (blue). The ANN architectures are (a),(b),(c): FFNN (d),(e),(f): LSTM. The reference sets are (a),(d): α-shape in $x_1 - x_2 - x_4$ (b),(e): α-shape in $x_2 - x_3 - x_4$ (c),(f): MPDF in $x_1 - x_2 - x_3 - x_4$.

B.5. Trajectory pruning

Figure B.12 and B.13 show the pruning result of the LSTM network trained with MPDF reference reachable sets of the double integrator and the inverted pendulum system. The same categorisation thresholds for scores are applied as presented in the scientific paper.

It can be observed for the inverted pendulum system the edges of the point clouds are classified as 'bad', with larger magnitudes of x_1. This may be caused by the choice of $p_c = 0.001$ being too high that has too much bias towards the corresponding regions in the state space. A lower choice of p_c may be more appropriate for the inverted pendulum, with relatively lower complexity compared to the quadrotor drone, to ensure that the 'safest' region according to the score is around the initial state.

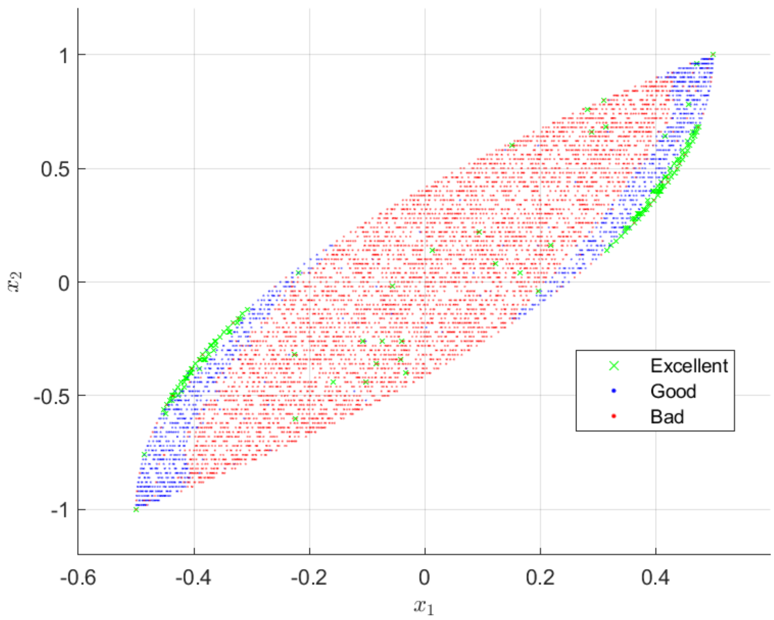


Figure B.12: MCS result on the double integrator system pruned using a LSTM trained with MPDF reference reachable set.

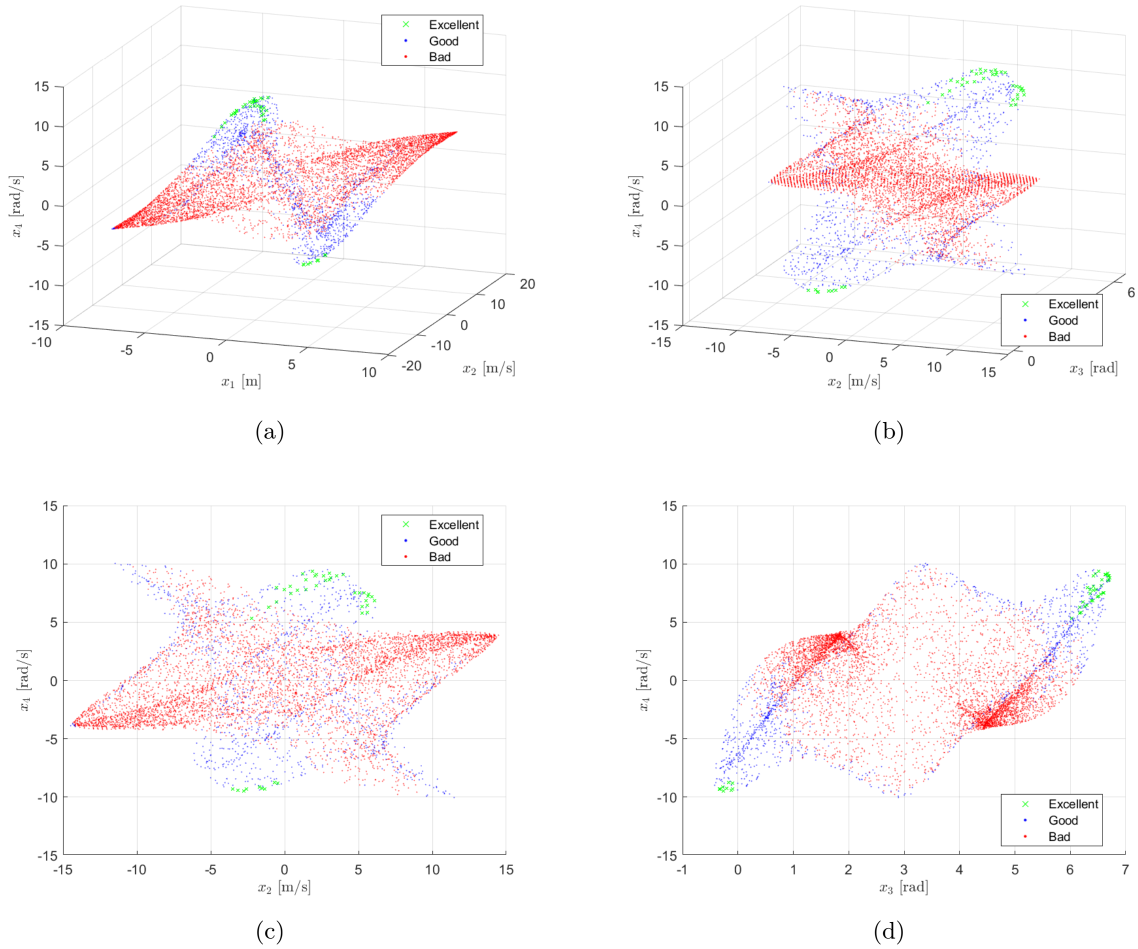


Figure B.13: MCS result on the inverted pendulum system pruned using a LSTM trained with MPDF reference reachable set, shown in three-dimensional projections.

B.6. Results for further discussions

Figure B.14 shows the pruning result of the MCS on the double integrator system using an ANN trained for the inverted pendulum system. Although a large portion of the 'Good' samples are near the reachable set boundary, most of the 'Excellent' samples are contained within the set. Furthermore the most extreme regions of the reachable sets are labelled as 'Bad'. This suggests that a valuable trajectory in one system may not be as valuable in another dynamic system, even though the inverted pendulum system is an extension of a double integrator (cart on wheel) system. Hence the introduction of the scores labelled to the end states allows to prune trajectories which a more general purely probabilistic approach cannot fully eliminate.

However it is also presented in the scientific paper that the trained network is able to prune trajectories with comparable accuracy when subject to model parameter changes. This means that there is a trade-off to be considered when applying this novel approach to off-nominal systems, that if the changes in the model parameters are too large the ANNs may not be useful in reducing MCS samples.

Figure B.15 shows 100 MCS end states of the double integrator system sampled with very high p_c of 0.8, in which duplicates of the constant inputs are rejected. A large portion of the end states coincide with the analytically derived reachable set. This is a simple yet effective way to sample more extreme control sequences.

With the current implementation of the ANN integrated to MCS-based reachability analysis, such simpler approaches may still be more efficient for a faster flight envelope estimation. However the high prediction power and the flexibility of ANN may see advantages over them when implementing this approach in for example a more adaptive manner or for more dynamically complex systems.

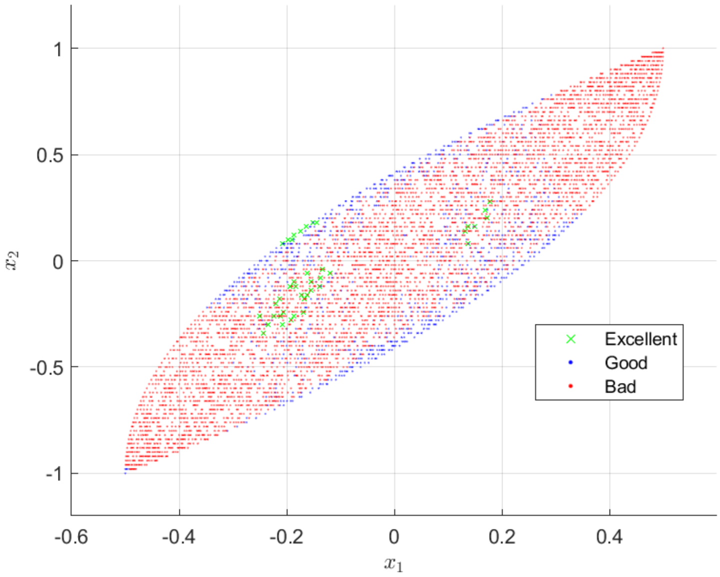


Figure B.14: MCS result on the double integrator system pruned using a LSTM trained with MPDF reference reachable set of the inverted pendulum system.

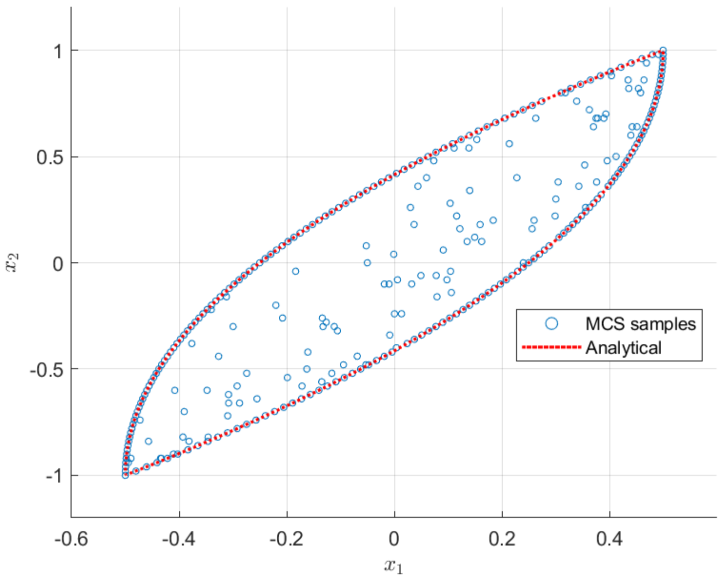


Figure B.15: 100 MCS end state samples using the double integrator problem with $p_c = 0.8$ while duplicates of the constant control sequences are ignored, shown together with the analytical reachable set.

References

[1] P. Lu and E. J. Van Kampen. “Active fault-tolerant control for quadrotors subjected to a complete rotor failure”. In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Vol. 2015-December. Institute of Electrical and Electronics Engineers Inc., 2015, pp. 4698–4703. DOI: 10.1109/IROS.2015.7354046.

[2] S. Sun et al. “Incremental Nonlinear Fault-Tolerant Control of a Quadrotor with Complete Loss of Two Opposing Rotors”. In: *IEEE Transactions on Robotics* 37.1 (Feb. 2021), pp. 116–130. DOI: 10.1109/TRO.2020.3010626.

[3] E. R. van Oort, Q. P. Chu, and J. A. Mulder. “Maneuver Envelope Determination through Reachability Analysis”. In: *Advances in Aerospace Guidance, Navigation and Control*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 91–102. ISBN: 978-3-642-19817-5.

[4] *Statistical Summary of Commercial Jet Airplane Accidents Worldwide Operations | 1959 – 2020*. Tech. rep. Seattle, Washington 98124-2207, USA: Boeing, 2021. URL: https://www.boeing.com/resources/boeingdotcom/company/about_bca/pdf/statsum.pdf.

[5] *A Statistical Analysis of Commercial Aviation Accidents 1958-2019*. Tech. rep. 31707 Blagnac Cedex, France: Airbus S.A.S., 2020. URL: https://accidentstats.airbus.com/sites/default/files/2021-03/Statistical-Analysis-of-Commercial-Aviation-Accidents-1958-2020.pdf.

[6] J. Wilborn and J. Foster. “Defining Commercial Transport Loss-of-Control: A Quantitative Approach”. In: *AIAA Atmospheric Flight Mechanics Conference and Exhibit*. Vol. 1. Reston, Virigina: American Institute of Aeronautics and Astronautics, Aug. 2004, pp. 205–215. ISBN: 978-1-62410-072-7. DOI: 10.2514/6.2004-4811. URL: https://arc.aiaa.org/doi/10.2514/6.2004-4811.

[7] W. Falkena et al. “Investigation of Practical Flight Envelope Protection Systems for Small Aircraft”. In: *Journal of Guidance, Control, and Dynamics* 34.4 (July 2011), pp. 976–988. ISSN: 0731-5090. DOI: 10.2514/1.53000. URL: https://arc.aiaa.org/doi/10.2514/1.53000.

[8] N. Tekles et al. “Design of a Flight Envelope Protection System for NASA’s Transport Class Model”. In: *Journal of Guidance, Control, and Dynamics* 40.4 (Apr. 2017), pp. 863–877. ISSN: 0731-5090. DOI: 10.2514/1.G001728. URL: https://arc.aiaa.org/doi/10.2514/1.G001728.

[9] L. Tang et al. “Methodologies for Adaptive Flight Envelope Estimation and Protection”. In: *AIAA Guidance, Navigation, and Control Conference*. Reston, Virigina: American Institute of Aeronautics and Astronautics, Aug. 2009. ISBN: 978-1-60086-978-5. DOI: 10.2514/6.2009-6260. URL: https://arc.aiaa.org/doi/10.2514/6.2009-6260.

[10] S. Kaynama et al. “Scalable Safety-Preserving Robust Control Synthesis for Continuous-Time Linear Systems”. In: *IEEE Transactions on Automatic Control* 60.11 (Nov. 2015), pp. 3065–3070. ISSN: 0018-9286. DOI: 10.1109/TAC.2015.2411872. URL: http://ieeexplore.ieee.org/document/7058350/.

[11] M. G. Goman, A. V. Khramtsovsky, and E. N. Kolesnikov. "Evaluation of Aircraft Performance and Maneuverability by Computation of Attainable Equilibrium Sets". In: *Journal of Guidance, Control, and Dynamics* 31.2 (Mar. 2008), pp. 329–339. ISSN: 0731-5090. DOI: 10.2514/1.29336. URL: https://arc.aiaa.org/doi/10.2514/1.29336.

[12] J. Lygeros. "On reachability and minimum cost optimal control". In: *Automatica* 40.6 (June 2004), pp. 917–927. ISSN: 00051098. DOI: 10.1016/j.automatica.2004.01.012. URL: https://linkinghub.elsevier.com/retrieve/pii/S0005109804000263.

[13] M. Lichter, A. Bateman, and G. Balas. "Flight Test Evaluation of a Run-Time Stability Margin Estimation Tool". In: *AIAA Guidance, Navigation, and Control Conference*. Reston, Virigina: American Institute of Aeronautics and Astronautics, Aug. 2009. ISBN: 978-1-60086-978-5. DOI: 10.2514/6.2009-6257. URL: https://arc.aiaa.org/doi/10.2514/6.2009-6257.

[14] M. Yin et al. "Probabilistic Flight Envelope Estimation with Application to Unstable Overactuated Aircraft". In: *Journal of Guidance, Control, and Dynamics* 42.12 (Dec. 2019), pp. 2650–2663. DOI: 10.2514/1.G004193. URL: https://arc.aiaa.org/doi/10.2514/1.G004193.

[15] S. Sun and C. C. de Visser. "Quadrotor Safe Flight Envelope Prediction in the High-Speed Regime: A Monte-Carlo Approach". In: *AIAA Scitech 2019 Forum*. Reston, Virginia: American Institute of Aeronautics and Astronautics, Jan. 2019. ISBN: 978-1-62410-578-4. DOI: 10.2514/6.2019-0948. URL: https://arc.aiaa.org/doi/10.2514/6.2019-0948.

[16] R. van den Brandt and C. de Visser. "Safe Flight Envelope Uncertainty Quantification using Probabilistic Reachability Analysis". In: *IFAC-PapersOnLine* 51.24 (2018), pp. 628–635. ISSN: 24058963. DOI: 10.1016/j.ifacol.2018.09.641. URL: https://linkinghub.elsevier.com/retrieve/pii/S240589631832353X.

[17] Y. Zhang et al. "Database-driven Safe Flight Envelope Protection for Impaired Aircraft". In: *AIAA Scitech 2020 Forum*. Vol. 1 PartF. Reston, Virginia: American Institute of Aeronautics and Astronautics, Jan. 2020, pp. 1–17. ISBN: 978-1-62410-595-1. DOI: 10.2514/6.2020-1374. URL: https://arc.aiaa.org/doi/10.2514/6.2020-1374.

[18] J. V. Foster and D. Hartman. "High-Fidelity Multi-Rotor Unmanned Aircraft System (UAS) Simulation Development for Trajectory Prediction Under Off-Nominal Flight Dynamics". In: *17th AIAA Aviation Technology, Integration, and Operations Conference*. Reston, Virginia: American Institute of Aeronautics and Astronautics, June 2017. ISBN: 978-1-62410-508-1. DOI: 10.2514/6.2017-3271. URL: https://arc.aiaa.org/doi/10.2514/6.2017-3271.

[19] R. Helsen et al. "Distance-Fields-Over-Grids Method for Aircraft Envelope Determination". In: *Journal of Guidance, Control, and Dynamics* 39.7 (July 2016), pp. 1470–1480. ISSN: 0731-5090. DOI: 10.2514/1.G000824. URL: https://arc.aiaa.org/doi/10.2514/1.G000824.

[20] J.-Y. Shin and C. Belcastro. "Robustness Analysis and Reliable Flight Regime Estimation of an Integrated Resilient Control System for a Transport Aircraft". In: *AIAA Guidance, Navigation and Control Conference and Exhibit*. Reston, Virigina: American Institute of Aeronautics and Astronautics, Aug. 2008. ISBN: 978-1-60086-999-0. DOI: 10.2514/6.2008-6656. URL: https://arc.aiaa.org/doi/10.2514/6.2008-6656.

[21] R. Pandita et al. "Reachability and Region of Attraction Analysis Applied to GTM Dynamic Flight Envelope Assessment". In: *AIAA Guidance, Navigation, and Control Conference*. Reston, Virigina: American Institute of Aeronautics and Astronautics, Aug. 2009. ISBN: 978-1-60086-978-5. DOI: 10.2514/6.2009-6258. URL: https://arc.aiaa.org/doi/10.2514/6.2009-6258.

[22] S. Osher and R. Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces*. Vol. 153. Applied Mathematical Sciences. New York, NY: Springer New York, 2003. ISBN: 978-1-4684-9251-4. DOI: 10.1007/b98879. URL: http://link.springer.com/10.1007/b98879.

[23] I. Kitsios and J. Lygeros. "Launch-Pad Abort Flight Envelope Computation for a Personnel Launch Vehicle Using Reachability". In: *AIAA Guidance, Navigation, and Control Conference and Exhibit*. Vol. 4. Reston, Virigina: American Institute of Aeronautics and Astronautics, Aug. 2005, pp. 3013–3023. ISBN: 978-1-62410-056-7. DOI: 10.2514/6.2005-6150. URL: https://arc.aiaa.org/doi/10.2514/6.2005-6150.

[24] J. Darbon and S. Osher. "Algorithms for overcoming the curse of dimensionality for certain Hamilton–Jacobi equations arising in control theory and elsewhere". In: *Research in the Mathematical Sciences* 3.1 (Dec. 2016), p. 19. ISSN: 2197-9847. DOI: 10.1186/s40687-016-0068-7. URL: http://link.springer.com/10.1186/s40687-016-0068-7.

[25] J. Darbon, P. M. Dower, and T. Meng. "Neural network architectures using min plus algebra for solving certain high dimensional optimal control problems and Hamilton-Jacobi PDEs". In: *arXiv preprint arXiv:2105.03336* (2021).

[26] E. Eyang, C. Combastel, and A. Zolghadri. "Determination of reachable flight regions: a zonotopic approach". In: *IFAC-PapersOnLine* 50.1 (July 2017), pp. 4014–4020. ISSN: 24058963. DOI: 10.1016/j.ifacol.2017.08.716. URL: https://linkinghub.elsevier.com/retrieve/pii/S240589631731162X.

[27] J. A. Sethian. *Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science*. Vol. 3. Cambridge university press, 1999.

[28] I. M. Mitchell. "Comparing Forward and Backward Reachability as Tools for Safety Analysis". In: *Hybrid Systems: Computation and Control*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 428–443. ISBN: 978-3-540-71493-4. URL: http://link.springer.com/10.1007/978-3-540-71493-4_34.

[29] M. Chen, J. C. Shih, and C. J. Tomlin. "Multi-vehicle collision avoidance via hamilton-jacobi reachability and mixed integer programming". In: *2016 IEEE 55th Conference on Decision and Control (CDC)*. IEEE, Dec. 2016, pp. 1695–1700. ISBN: 978-1-5090-1837-6. DOI: 10.1109/CDC.2016.7798509. URL: http://ieeexplore.ieee.org/document/7798509/.

[30] H. Seo et al. "Robust Trajectory Planning for a Multirotor against Disturbance based on Hamilton-Jacobi Reachability Analysis". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Nov. 2019, pp. 3150–3157. ISBN: 978-1-7281-4004-9. DOI: 10.1109/IROS40897.2019.8968126. URL: https://ieeexplore.ieee.org/document/8968126/.

[31] T. Lombaerts et al. “Safe Maneuvering Envelope Estimation based on a Physical Approach”. In: *AIAA Guidance, Navigation, and Control (GNC) Conference*. Reston, Virginia: American Institute of Aeronautics and Astronautics, Aug. 2013. ISBN: 978-1-62410-224-0. DOI: 10.2514/6.2013-4618. URL: https://arc.aiaa.org/doi/10.2514/6.2013-4618.

[32] I. M. Mitchell. “The Flexible, Extensible and Efficient Toolbox of Level Set Methods”. In: *Journal of Scientific Computing* 35.2-3 (June 2008), pp. 300–329. ISSN: 0885-7474. DOI: 10.1007/s10915-007-9174-4. URL: http://link.springer.com/10.1007/s10915-007-9174-4.

[33] Y. Zhang. “Database-driven online safe flight envelope prediction and protection for enhanced aircraft fault tolerance”. PhD thesis. Delft University of Technology, 2019. ISBN: 978-94-028-1418-7. DOI: 10.4233/uuid:70778c5c-3f00-4854-a0fb-98b24c9ed1cb.

[34] J. Stapel et al. “Efficient Methods for Flight Envelope Estimation through Reachability Analysis”. In: *AIAA Guidance, Navigation, and Control Conference*. Reston, Virginia: American Institute of Aeronautics and Astronautics, Jan. 2016. ISBN: 978-1-62410-389-6. DOI: 10.2514/6.2016-0083. URL: https://arc.aiaa.org/doi/10.2514/6.2016-0083.

[35] R. Bellman. “Dynamic programming and stochastic control processes”. In: *Information and Control* 1.3 (Sept. 1958), pp. 228–239. ISSN: 00199958. DOI: 10.1016/S0019-9958(58)80003-0. URL: https://linkinghub.elsevier.com/retrieve/pii/S0019995858800030.

[36] M. Althoff and B. H. Krogh. “Zonotope bundles for the efficient computation of reachable sets”. In: *IEEE Conference on Decision and Control and European Control Conference*. IEEE, Dec. 2011, pp. 6814–6821. ISBN: 978-1-61284-801-3. DOI: 10.1109/CDC.2011.6160872. URL: http://ieeexplore.ieee.org/document/6160872/.

[37] N. Kochdumper and M. Althoff. “Sparse Polynomial Zonotopes: A Novel Set Representation for Reachability Analysis”. In: *IEEE Transactions on Automatic Control* 66.9 (Sept. 2021), pp. 4043–4058. ISSN: 0018-9286. DOI: 10.1109/TAC.2020.3024348. URL: https://ieeexplore.ieee.org/document/9199088/.

[38] M. Althoff, O. Stursberg, and M. Buss. “Computing reachable sets of hybrid systems using a combination of zonotopes and polytopes”. In: *Nonlinear Analysis: Hybrid Systems* 4.2 (May 2010), pp. 233–249. ISSN: 1751570X. DOI: 10.1016/j.nahs.2009.03.009. URL: https://linkinghub.elsevier.com/retrieve/pii/S1751570X09000442.

[39] G. Delansnay et al. “Zonotopic Reachability Analysis of Multirotor Aircraft”. In: *2021 25th International Conference on System Theory, Control and Computing (ICSTCC)*. IEEE, Oct. 2021, pp. 576–581. ISBN: 978-1-6654-1496-8. DOI: 10.1109/ICSTCC52150.2021.9607056. URL: https://ieeexplore.ieee.org/document/9607056/.

[40] H. G. Harno and Y. Kim. “Flight envelope estimation for helicopters under icing conditions via the zonotopic reachability analysis”. In: *Aerospace Science and Technology* 102 (July 2020), p. 105859. ISSN: 12709638. DOI: 10.1016/j.ast.2020.105859. URL: https://linkinghub.elsevier.com/retrieve/pii/S1270963820305411.

[41] M. Chen et al. “Decomposition of Reachable Sets and Tubes for a Class of Nonlinear Systems”. In: *IEEE Transactions on Automatic Control* 63.11 (Nov. 2018), pp. 3675–3688. ISSN: 0018-9286. DOI: 10.1109/TAC.2018.2797194. URL: https://ieeexplore.ieee.org/document/8267187/.

[42] M. Chen, S. Herbert, and C. J. Tomlin. "Fast reachable set approximations via state decoupling disturbances". In: *2016 IEEE 55th Conference on Decision and Control (CDC)*. IEEE, Dec. 2016, pp. 191–196. ISBN: 978-1-5090-1837-6. DOI: 10.1109/CDC.2016.7798268. URL: http://ieeexplore.ieee.org/document/7798268/.

[43] D. Lee and C. J. Tomlin. "Iterative Method using the Generalized Hopf Formula: Avoiding Spatial Discretization for Computing Solutions of Hamilton-Jacobi Equations for Nonlinear Systems". In: *2019 IEEE 58th Conference on Decision and Control (CDC)*. Vol. 2019-December. IEEE, Dec. 2019, pp. 1486–1493. ISBN: 978-1-7281-1398-2. DOI: 10.1109/CDC40024.2019.9029572. URL: https://ieeexplore.ieee.org/document/9029572/.

[44] N. H. Campbell, M. J. Acheson, and I. M. Gregory. "Dynamic Vehicle Assessment for Intelligent Contingency Management of Urban Air Mobility Vehicles". In: *AIAA Scitech 2021 Forum*. Reston, Virginia: American Institute of Aeronautics and Astronautics, Jan. 2021, pp. 1–15. ISBN: 978-1-62410-609-5. DOI: 10.2514/6.2021-1001. URL: https://arc.aiaa.org/doi/10.2514/6.2021-1001.

[45] M. Raissi, P. Perdikaris, and G. Karniadakis. "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations". In: *Journal of Computational Physics* 378 (Feb. 2019), pp. 686–707. ISSN: 00219991. DOI: 10.1016/j.jcp.2018.10.045. URL: https://linkinghub.elsevier.com/retrieve/pii/S0021999118307125.

[46] J. Darbon, G. P. Langlois, and T. Meng. "Overcoming the curse of dimensionality for some Hamilton–Jacobi partial differential equations via neural network architectures". In: *Research in the Mathematical Sciences* 7.3 (Sept. 2020), p. 20. ISSN: 2522-0144. DOI: 10.1007/s40687-020-00215-6. URL: https://link.springer.com/10.1007/s40687-020-00215-6.

[47] S. Mahadevan. "Monte carlo simulation". In: *Reliability-Based Mechanical Design*. Ed. by T. A. Cruse. 1st ed. CRC Press, 1997. Chap. 4, pp. 123–146. ISBN: 9780824797935.

[48] G. R. Terrell and D. W. Scott. "Variable Kernel Density Estimation". In: *The Annals of Statistics* 20.3 (Sept. 1992), pp. 1236–1265. ISSN: 0090-5364. DOI: 10.1214/aos/1176348768. URL: https://projecteuclid.org/journals/annals-of-statistics/volume-20/issue-3/Variable-Kernel-Density-Estimation/10.1214/aos/1176348768.full.

[49] S. Sun, C. C. de Visser, and Q. Chu. "Quadrotor Gray-Box Model Identification from High-Speed Flight Data". In: *Journal of Aircraft* 56.2 (Mar. 2019), pp. 645–661. ISSN: 0021-8669. DOI: 10.2514/1.C035135. URL: https://arc.aiaa.org/doi/10.2514/1.C035135.

[50] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. URL: http://www.deeplearningbook.org.

[51] S. Shalev-Shwartz and S. Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014. ISBN: 1107057132. URL: http://www.cs.huji.ac.il/~shais/UnderstandingMachineLearning.

[52] X.-S. Yang. "Genetic Algorithms". In: *Nature-Inspired Optimization Algorithms*. Elsevier, Jan. 2021, pp. 91–100. DOI: 10.1016/B978-0-12-821986-7.00013-5. URL: https://linkinghub.elsevier.com/retrieve/pii/B9780128219867000135.

[53] G. Cybenko. "Approximation by superpositions of a sigmoidal function". In: *Mathematics of Control, Signals, and Systems* 2.4 (Dec. 1989), pp. 303–314. ISSN: 0932-4194. DOI: 10.1007/BF02551274. URL: http://link.springer.com/10.1007/BF02551274.

[54] *Renewable Energy Asset Management software and services*. Dec. 2021. URL: url:https://skyspecs.com/.

[55] *THE DRONE RACING LEAGUE*. Dec. 2021. URL: https://thedroneracingleague.com/.

[56] B. J. Emran and H. Najjaran. "A review of quadrotor: An underactuated mechanical system". In: *Annual Reviews in Control* 46 (Jan. 2018), pp. 165–180. ISSN: 1367-5788. DOI: 10.1016/J.ARCONTROL.2018.10.009.

[57] A. Das, F. Lewis, and K. Subbarao. "Backstepping Approach for Controlling a Quadrotor Using Lagrange Form Dynamics". In: *Journal of Intelligent and Robotic Systems 2009 56:1* 56.1 (Apr. 2009), pp. 127–151. ISSN: 1573-0409. DOI: 10.1007/S10846-009-9331-0. URL: https://link.springer.com/article/10.1007/s10846-009-9331-0.

[58] E. J. Smeur, G. C. de Croon, and Q. Chu. "Gust disturbance alleviation with Incremental Nonlinear Dynamic Inversion". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Vol. 2016-November. 2016, pp. 5626–5631. ISBN: 978-1-5090-3762-9. DOI: 10.1109/IROS.2016.7759827. URL: http://ieeexplore.ieee.org/document/7759827/.

[59] Y. Zhong et al. "Robust Actuator Fault Detection and Diagnosis for a Quadrotor UAV With External Disturbances". In: *IEEE Access* 6 (2018), pp. 48169–48180. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2018.2867574. URL: https://ieeexplore.ieee.org/document/8453038/.

[60] 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*. Vol. 6. 2011, pp. 1147–1151. URL: https://ieeexplore.ieee.org/document/6106100.

[61] M Athans and P. L. Falb. *Optimal Control*. McGraw Hill, 1966.

[62] A. Adardour. *Nonlinear Reachability Analysis using Multivariate B-splines*. Nov. 2016. URL: https://repository.tudelft.nl/islandora/object/uuid%3A19aab69b-3f58-4519-82e8-20a43146666f.

[63] F. Chollet and others. *Keras*. 2015. URL: https://keras.io.