Optimisation-Based Verification Process of Obstacle Avoidance Systems for Unmanned Vehicles

Sivaranjini Thedchanamoorthy

Department of Aeronautical and Automotive Engineering

Loughborough University

A thesis submitted for the degree of

Doctor of Philosophy

September 2013

In memory of my Brother....

Acknowledgements

I would like to express my gratitude to my supervisor, Prof. Wen-Hua Chen, for his continuing support, encouragement and guidance throughout my research study. I wish to thank him for his patience, supervision and advice through my research and during the completion of this thesis.

I would like to thank all of my colleagues in Control and Reliability group for their support and for sharing their knowledge.

I am thankful for the fully funded studentship from the Department of Aeronautical and Automotive Engineering, Loughborough University.

Finally, I am deeply grateful to my family and my husband for all support they gave me to reach this point.

Abstract

This thesis deals with safety verification analysis of collision avoidance systems for unmanned vehicles. The safety of the vehicle is dependent on collision avoidance algorithms and associated control laws, and it must be proven that the collision avoidance algorithms and controllers are functioning correctly in all nominal conditions, various failure conditions and in the presence of possible variations in the vehicle and operational environment. The current widely used exhaustive search based approaches are not suitable for safety analysis of autonomous vehicles due to the large number of possible variations and the complexity of algorithms and the systems. To address this topic, a new optimisation-based verification method is developed to verify the safety of collision avoidance systems.

The proposed verification method formulates the worst case analysis problem arising the verification of collision avoidance systems into an optimisation problem and employs optimisation algorithms to automatically search the worst cases. Minimum distance to the obstacle during the collision avoidance manoeuvre is defined as the objective function of the optimisation problem, and realistic simulation consisting of the detailed vehicle dynamics, the operational environment, the collision avoidance algorithm and low level control laws is embedded in the optimisation process. This enables the verification process to take into account the parameters variations in the vehicle, the change of the environment, the uncertainties in sensors, and in particular the mismatching between model used for developing the collision avoidance algorithms and the real vehicle. It is shown that the resultant simulation based optimisation problem is non-convex and there might be many local optima.

To illustrate and investigate the proposed optimisation based verification process, the potential field method and decision making collision avoidance method are chosen as an obstacle avoidance candidate technique for verification study. Five benchmark case studies are investigated in this thesis: static obstacle avoidance system of a simple unicycle robot, moving obstacle avoidance system for a Pioneer 3DX robot, and a 6 Degrees of Freedom fixed wing Unmanned Aerial Vehicle with static and moving collision avoidance algorithms. It is proven that although a local optimisation method for nonlinear optimisation is quite efficient, it is not able to find the most dangerous situation. Results in this thesis show that, among all the global optimisation methods that have been investigated, the DIviding RECTangle method provides most promising performance for verification of collision avoidance functions in terms of guaranteed capability in searching worst scenarios.

Contents

List of Figures

List of Tables

Abbreviations

OAS -Obstacle Avoidance System

UGV- Unmanned Ground Vehicle

UAV- Unmanned Aerial Vehicle

V&V - Verification and Validation

FFA- Functional Failure Analysis

DoF-Degrees of Freedom

MCM - Monte Carlo method

FTA - Fault Tree Analysis

HAZOP - Hazard and Operability Studies

FMEA - Failure Modes and Effects Analysis

TCAS- Traffic Alert and Collision Avoidance Systems

RRT -Rapidly-Exploring Random Trees

CD&R - Conflict Detection and Resolution

DIRECT - DIvidingRECTangles

GA – Genetic Algorithm

FAA - Federal Aviation Authority

WWR - Wheel Mobile Robot

c-space - configuration space

PRM - Probabilistic RoadMap

RRT - Rapidly Exploring Random Tree

ACA - Autonomous Collision Avoidance

SQP - Sequential Quadratic Programming

BFGS - Broyden-Fletcher-Goldfarb-Shanno

Chapter 1

Introduction

1.1 Overview

The integration of autonomous vehicles requires new methods to certify Obstacle Avoidance System (OAS). Motion safety for autonomous vehicles is one of the main barriers preventing a much wider application of unmanned vehicles. Path planning and navigation schemes aim at guiding unmanned vehicles reaching a goal safely while avoiding collision in known/unknown environments. To this end, many collision avoidance algorithms have been proposed and tried on various applications. However, it is still far away to prove that such algorithms are reliable and always provide adequate performance under all possible events in real operation. In addition to offering better performance, a key practical concern related to any new method is to reduce the risk of collisions in the presence of all possible parameter variations and various failure conditions. Therefore, all proposed collision avoidance algorithms have to be verified under all operational conditions and variations that may be experienced during the life of unmanned vehicles. The objective of this project is to develop advanced algorithms to support the deployment of safety-critical OAS for unmanned vehicles.

This thesis develops novel optimisation-based methods for worst-case analysis of OAS for unmanned vehicles in the presence of uncertainties and variations. The optimisation-based verification method is applied to the OAS and identified the most critical situations in the presence of uncertainties. The verification of OAS for Unmanned Ground Vehicles (UGVs) and Unmanned Aerial Vehicles (UAVs) are considered in this thesis. The minimum distance to the obstacle is defined as the objective function subject to uncertain parameters lower and upper bounds. Local and global optimisation-based verification processes are developed to automatically search the worst combinations of the parameters and the worst-case distance between the vehicle and an obstacle under all possible variations and uncertainties. Different optimisation algorithms are applied and compared in terms of the reliability and efficiency. Moreover, the Monte Carlo simulation is carried out to provide a benchmark comparison of the proposed automatic worst-case search methods.

The thesis is organized as follows: This chapter first begins with an introduction to the safety-critical problem for unmanned vehicles and recent research in this area. After that, the proposed novel methodology for verifying the OAS for unmanned vehicles is introduced. The final section of this chapter contains relevant recent research of collision avoidance algorithms. Chapter 2 provides the definition of Verification and Validation (V&V) and brief introduction of static and moving obstacle avoidance algorithms. Furthermore, Chapter 2 gives the conclusion to select the obstacle avoidance algorithms for this verification of OAS studies.

Chapter 3 discusses the robustness analysis of OAS, and four case studies are considered. In order to understand the novel V&V algorithm of OAS, first very simple unicycle robot is considered, and only two uncertain parameters are selected within the lower and upper bounds. The artificial potential field method is chosen as a path planning and obstacle avoidance candidate technique for verification study for static obstacles in 2D environment. Then this work is extended to more complex unicycle Pioneer 3DX robot, and moving obstacle avoidance algorithms are developed in 2D to verify the OAS. Furthermore, based on a 6 Degree of Freedom (6DoF) kinematic and dynamic model of UAV, the path planning and collision avoidance algorithms for static and moving obstacles using with potential field method are developed in 3D space. Uncertainty analysis is investigated to select the most significant parameters.

Chapter 4 gives the local optimisation-based verification of OAS for unmanned vehicles. It is shown that local optimisation-based approach may fail to find the worst cases. To overcome this problem, two stochastic global optimisation algorithm based verification processes are developed in Chapter 5. Furthermore, in order to guarantee finding the worst cases, a deterministic global optimisation method is applied to OAS. Global optimisation results show that the collision avoidance algorithm functions correctly in the parameter variations.

Chapter 6 describes the Monte Carlo method (MCM) on computer software for generation of random variations of minimum distance to the obstacle. Chapter 7 discusses the safety analysis of decision making algorithms for UAVs. Chapter 8 provides some conclusions and discusses future direction for the research described in the thesis.

The research described in this thesis is based on the following articles, which have been published for publication [A.1. 1 - 6].

1.2 Safety-critical Problem for Unmanned Vehicles

V&V techniques are essential in any safety critical systems. In the development of OAS for unmanned vehicles, it is important to identify if the system meets the requirements and achieves the goal without any collisions under all parameter variations and failure conditions. This is the process of V&V of OAS, and it consists of planning from the start of vehicle model, the development of collision avoidance algorithms and several tests. This V&V process must be performed before doing the testing of unmanned vehicles. After the testing, the results should be further analyzed to validate the results. Therefore, V&V of OAS is a very expensive and time consuming process, especially for fighter aircraft, where many different combinations of aircraft parameters must be investigated involving large variations of mass, inertia, and centre-of-gravity location, as well as uncertain aerodynamic data [1]. Fig.1.1 shows a schematic of a Tornado aircraft carrying a heavy store load [2]. The potential variation in aircraft mass, inertia and centre-of-gravity, due to the carriage and release of such stores is obvious. Therefore, verifying OAS of unmanned vehicles becomes more important and challenging problem in the development process of autonomous vehicles.

Requirements, design, and test coverage and their quantification all significantly impact overall system quality, but control law software test coverage is especially significant to development costs. For current flight-safety-critical systems, control law, software implementation, and test comprise over 60% of total development costs (Fig.1.2) [3]. This percentage will be even higher using V&V strategies on military aerospace safety critical systems.

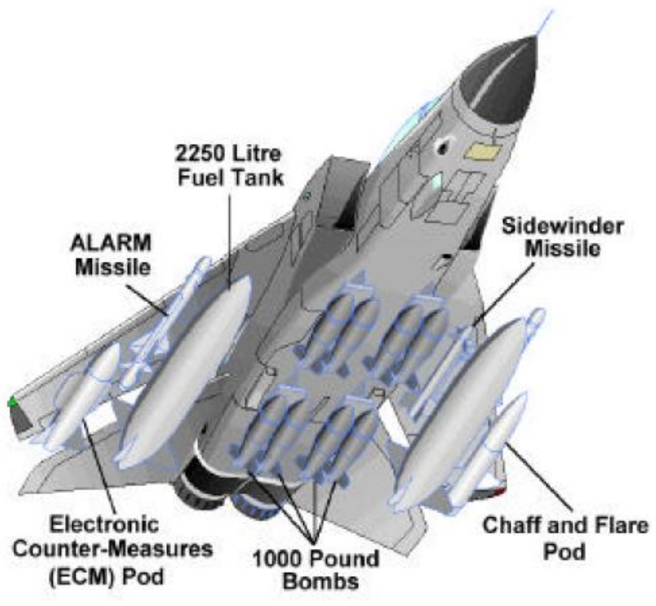


Figure 1.1: Stores carriage [2]

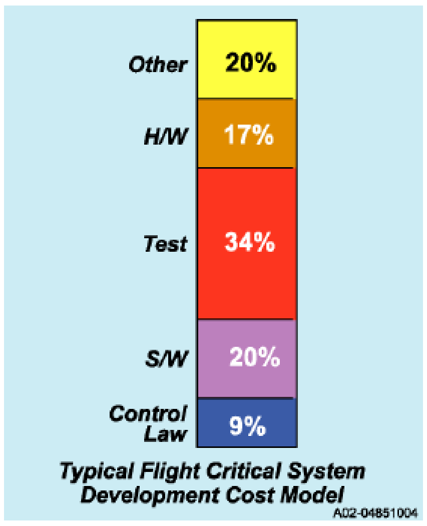


Figure 1.2: Flight Critical System Cost Model [3]

Three major concerns in regard to autonomous vehicle operation are efficiency, safety and accuracy. As the safety of the vehicle is dependent on the controller and the collision avoidance algorithm, it must be proven that the controller and collision avoidance algorithm function correctly in the presence of all possible vehicle and environmental variations. Two particular difficulties faced by designers are a mismatching between the model used for algorithm development and the real vehicle dynamics and various uncertainties in vehicle operations. To simplify the process of the algorithm development, in general a much simplified model that can capture the main characters of the vehicle is used in the design stage under a number of assumptions or simplifications. The variations of the autonomous vehicle dynamics in operation may arise due to the changes of the vehicle itself (e.g. the change of mass or the centre of gravity) or the change of the operation environment (e.g. tyre friction for different road surfaces, wind velocity). The verification process is to prove that the vehicle is safe under all conditions and variations. This means that the work must be performed not only for the nominal model, but also for all possible vehicle and environmental variations. Therefore, the collision avoidance algorithm has to be extended to analyze of such uncertainties. All possible combinations of vehicle parameters must be investigated. Before the first vehicle manoeuvre can be executed, the verification process of OAS must be performed to prove that the controlled vehicle meets all the clearance criteria. This is particularly important for safety critical functions such as collision avoidance.

In the early stages of the safety analysis, the well-established classical safety analysis techniques such as Functional Failure Analysis (FFA), Fault Tree Analysis (FTA), Hazard and Operability Studies (HAZOP) and Failure Modes and Effects Analysis (FMEA) were used in the robot industry [4]. In [7], the FTA method was applied to Bremen autonomous wheelchair system for the safety analysis. FTA method was also applied to the TCAS (Traffic Alert and Collision Avoidance Systems) for the safety analysis in [8]. These techniques are still widely practiced in safety assessments. As the complexity of modern programmable electronic systems increases, however, the application of classical techniques is becoming increasingly more problematic. Several modifications have been discussed to overcome this problem [4 - 6]. In [9], the common Unified Modelling Language (UML) method was applied to the robot for the safety analysis, and risk analysis was then performed using a Preliminary Hazard analysis, an adaption of the HAZOP method and the classical FTA.

Reachability analysis methods have been used in verification of hybrid systems [10 ; 11]. If the approximated set of reachable states has no intersection with the unsafe set, then the system is safe in these algorithms. However, this algorithm is constrained by computational complexity. A collision avoidance manoeuvre is developed with smooth, fully flyable curves in [12]. Then, verification algorithm for logic of hybrid systems is applied to this collision avoidance in the flyable tangential roundabout manoeuvre. However, in this verification method, manual effort is still needed to simplify arithmetical complexity and modularize the proof appropriately. The reachable sets are computed in [13] to verify the safety of autonomous car as well as moving objects in its environment. This method has been developed for hybrid systems. In order to speed up the verification process for online application, continuous system dynamics of the cognitive car and other traffic participants are abstracted by Markov chains.

In [14], the methodology is developed for computing reachable sets for hybrid systems and the corresponding methods for computing controllers that guarantee safety and target capture. This method is used to verify the behaviour of safety-critical dynamical systems and design control laws for such systems with verified behaviour. A testing method for safety/reachability analysis of stochastic hybrid systems is proposed in [15]. Testing based method is very appealing because of the simplicity of its execution, the possibility of having a partial verification and its highly parallel structure. In [16], a methodology for safety verification of continuous and hybrid systems in the worst-case and stochastic settings are presented. In the worst-case setting, a function of state termed barrier-certificate is used to certify that all trajectories of the system starting from a given initial set do not enter an unsafe region. To search for a barrier-certificate, the non-convex set cannot be performed through SOS (Sum of Squares) optimisation. Therefore, appropriate sum of squares conditions are formulated, but solving a non-convex optimisation problem by iteration is not guaranteed to yield the globally optimal solution. In [17], a sampling-based verification algorithm based on Resolution Complete falsification is proposed. Sufficient conditions that guarantee resolution completeness are derived. However, these conditions are conservative and require a high-resolution sampling in state and input spaces for most practical problems.

Thierry Fraichard [18] proposed three safety criteria for the safety analysis of mobile robotics systems, and a number of existing collision avoidance schemes are evaluated with respect to these three safety criteria. It has been established that in all cases Nearness Diagram, Dynamic Window, and Velocity Obstacle violated one or several of the safety criteria. Motion safety of these approaches, especially in the presence of moving objects, could not be guaranteed. The safety analysis also shows that only the Inevitable Collision States method satisfies these three safety criteria.

1.3 A novel V&V algorithm of OAS for Unmanned vehicles

Analysis cycle used for the verification of an obstacle avoidance system is illustrated by the flow diagram in Fig.1.3. Initialisation is the first step for the obstacle avoidance verification process. Anti-collision condition is defined and uncertain parameters are chosen to determine the worst-case. Before applying the optimisation algorithm, the anti-collision condition can be checked for nominal case. If it is satisfied, then the optimisation methods are applied to determine the worst-case condition and worst-case parameters. Otherwise, the obstacle avoidance algorithm and controller have to be redefined to satisfy the anti-collision condition. More details of this analysis cycle are described below:

Step 1: Generation of an analysis model and obstacle avoidance algorithms

The first task of the verification process of OAS is the investigation of the vehicle model and collision avoidance algorithms. For a design purpose a reduced model is often used, but the verification process work requires a full-size nonlinear vehicle model which includes all parametric uncertainties. From this nonlinear model, linear models are derived and these linear models have to be validated against the nonlinear model. In the development of collision avoidance algorithms, only a kinematic model of the vehicle is used in the obstacle avoidance method (see Fig.1.4). This greatly simplifies the analysis and design of the collision avoidance algorithms. However, the model in the verification stage must be as close to the real world as possible, which demands a much more complicated model including kinematic and dynamic model, the speed controller and motion controller (see Fig.1.5).

The goal of motion planning is to generate the desired trajectory to be fed into the motion controller so that the vehicle tracks the desired trajectory. Figs.1.4-1.5 illustrate how this functionality can be implemented for autonomous vehicles. Waypoints information supplies to the motion control system, and then motion planner retrieves the waypoints, generates a desired trajectory which includes the desired velocity. These desired velocities are fed into the speed controller to obtain a control command.

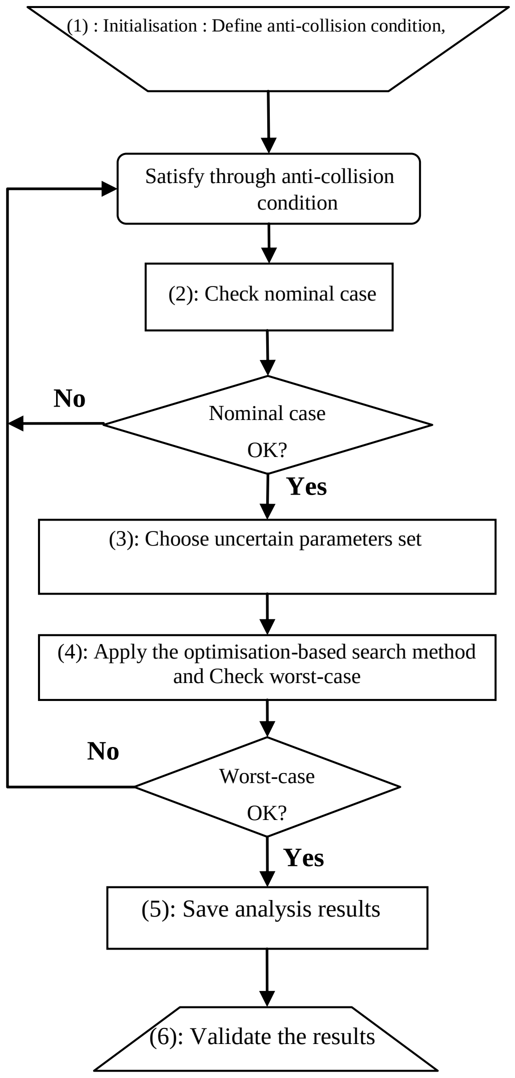

(1) : Initialisation : Define anti-collision condition,
Satisfy through anti-collision condition
(2): Check nominal case
No
Nominal case OK?
Yes
(3): Choose uncertain parameters set
(4): Apply the optimisation-based search method and Check worst-case
No
Worst-case OK?
Yes
(5): Save analysis results
(6): Validate the results

Figure 1.3: Optimisation-based verification analysis cycle for OAS

In a work environment, there are two types of obstacles: static obstacles and moving obstacles. Robot has to find a collision-free path between the starting point and the goal in an environment containing various static and moving obstacles. To assess the safety of vehicles, the minimum distance to the obstacle *(d_{min})* is defined as the objective function in the time domain.

$$
\begin{aligned}
d_{min} = min(d(t)) \quad &\text{for } t \le T\ (sec) \\
\text{s.t} \quad &P_L \le P \le P_U
\end{aligned}
\tag{1.1}
$$

where P is the uncertain parameters set; P_L and P_U are lower and upper bounds of P. T is the time period of the collision avoidance manoeuvre and $d(t)$ is the distance to the obstacle and calculated using simulation with the completed model of the vehicle in Fig.1.5.

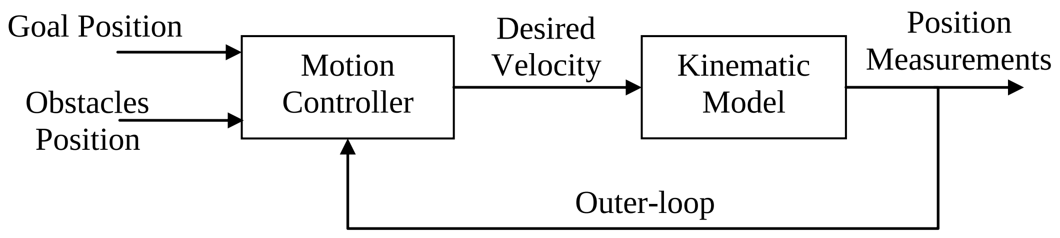


Figure 1.4: Simplified model for obstacle avoidance algorithm development

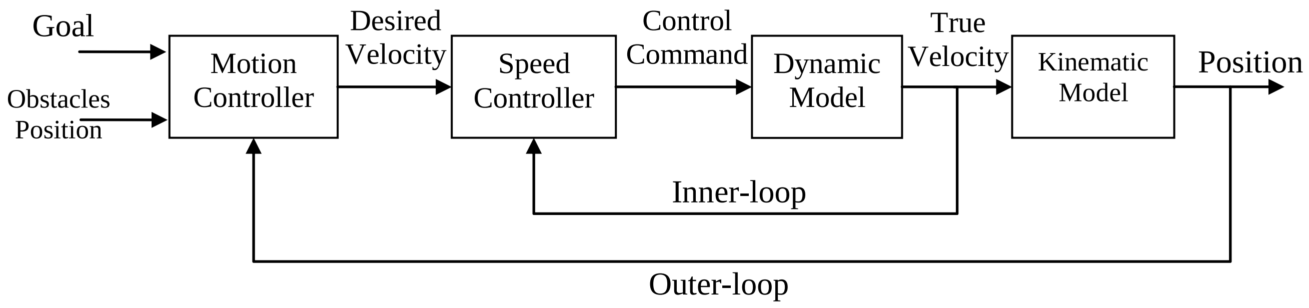


Figure 1.5: Obstacle avoidance systems

According to the obstacles' shapes, the anti-collision condition is initialized to verify the OAS. The specified safe margin can be chosen according to the vehicle's dimensions. For example, in Fig.1.6 for a circular static obstacle, letting $r = r_0 + r_{safe}$, the anti-collision condition is defined as

$$d_{min} > r \tag{1.2}$$

where r is the radius of circular obstacle and r_{safe} is the safe margin. Many collision avoidance algorithms have been developed in the last three decades. These obstacle avoidance algorithms can be applied to the vehicle model, and these algorithms have to be verified under all parameters variations.

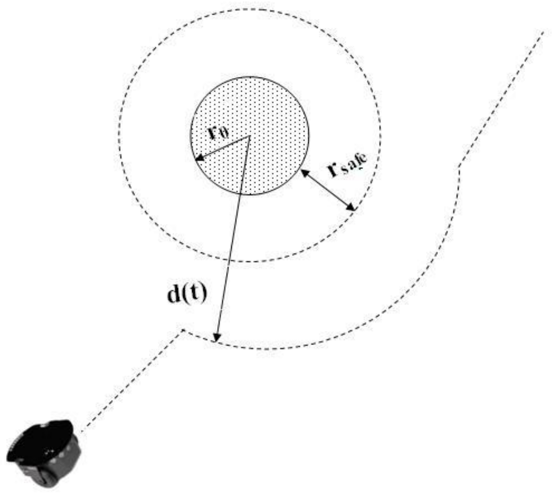


Figure 1.6: Obstacle avoidance clearance criterion

Step 2: Analysis at nominal case

After choosing the appropriate speed controller and motion controller, the anti-collision condition is checked at nominal parameters. If it is satisfied, then the next step of initial robustness analysis will be considered. Otherwise, the controller and obstacle avoidance algorithm will be redefined to satisfy the anti-collision condition. Therefore, this is to confirm that a desirable performance is achieved at the nominal case under the design described in the step 1.

Step 3: Studies on the effect of uncertainties

For clearance, it must be demonstrated that the vehicle is safe under all conditions and variations. This means that the assessment must be performed not only for the nominal model, but also for all possible parameters variations, operating conditions and failure conditions.

The types of uncertainties are given below [1]:

- Configuration dependent variability- (e.g. mass, inertia and centre of gravity which differ with stores and fuel)
- Aerodynamic uncertainties –(on stability derivatives, control power and damping derivatives)
- Hardware dependant variability- (changes of the actuator or sensor dynamics or computing delays)
- Air data system dependant tolerances-(measurement errors in signals like angle of attack, Mack number or dynamic pressure which are used for scheduling of the control laws).

Uncertainties are introduced in the parameters to allow varying within ±10% or ±20% of its nominal value. The parameters variations versus minimum distance to the obstacle responses are analysed to select the most significant parameters. Furthermore, whether it is a linear or nonlinear analysis is predicted according to the parameters variation versus minimum distance to the obstacle plots.

Step 4: Worst-case analysis

After choosing the objective function (Eq.1.1) subject to uncertain parameters bounds, the optimisation algorithms are applied to the OAS to find the worst-case condition and worst-case parameters set. Moreover, these worst case results must be guaranteed by the optimisation algorithm. It is a bound constrained linear or nonlinear optimisation problem and challenging problem. If anti-collision condition is not satisfied at worst-case parameters, then the verification process will be carried out from the beginning.

Step 5: Save the analysis results

The anti-collision condition is checked at worst-case parameters. If it is satisfied at worst-case parameters, then the worst-case condition and worst-case parameter are stored and the validation process will be carried out.

Step 6: Validate the worst-case results

These worst-case condition and worst-case parameters identified in the verification process are further validated using with simulation responses.

1.4 Overview of Obstacle avoidance algorithms

Autonomous vehicles have been used to perform in both civil and military applications such as reconnaissance, environmental monitoring, border patrol, search and rescue operations, disaster relief, traffic monitoring etc. Without a pilot, computer algorithms must be developed to generate a feasible path in real time. Depending on the operation scenarios, there are different kinds of path planning methods. Several algorithms have been developed for mobile robot path planning in the presence of known obstacles. These algorithms include Roadmap, Cell-Decomposition and Potential Field methods [55]. Potential field method is widely used for mobile robot obstacle avoidance path planning for both static and dynamic environments [80]. The well-known Roadmaps methods are Visibility graph, Voronoi diagram, Probabilistic RoadMaps and Rapidly-Exploring Random Trees (RRT) [55]. In [21; 61], closed-loop RRT algorithm is applied to the real-time motion planning.

There are many other intelligent algorithms for path planning including Genetic Algorithm [22], Ant Colony Algorithm [23], Neural Networks [24], Fuzzy Logic [25]. However, these algorithms cannot reach an ideal solution in complex dynamic environment [26]. Raja et al [27] developed an obstacle avoidance algorithm for convex polygonal and curved obstacles with an objective of minimizing travelling distance and computational time in static environment. However, this algorithm is not suitable for concave polygonal obstacles and moving obstacles. Sipahioglu et al [28] proposed real-time tour construction for a mobile robot in a dynamic environment. A heuristic-based travelling salesman problem is applied, and Dijsktra algorithm is used to determine the feasible tour. In [29], a global path planning algorithm developed for a robot moving in an environment cluttered with obstacles which have arbitrary shape, size and location. Static obstacles are negotiated by using ‘direction concept’ and moving obstacles are by ‘waiting time concept’ algorithms. These algorithms can be applied to the environments having any irregular shape obstacles.

The combined method of ray tracing and limit cycle path planning algorithm is proposed in [30] for UAV in both 2D and 3D space. Griffiths et al [31] presented a RRT based path planner through 3D environment for an autonomous aerial vehicle. In [32], both probabilistic roadmap-based and RRT algorithms are used for generating 3D collision free path for an autonomous helicopter. Bortoff [33] developed a collision free path planning method using Voronoi graph search method, whereas a model predicative control based trajectory optimisation method is used to avoid obstacles in [34]. UAV motion planning techniques based on potential field functions have been extensively studied; e.g. [35, 36]. Collision avoidance scheme for UAVs based on Proportional Navigation (PN) guidance is presented in [37]. A vision based Grossberg Neural Network (GNN) scheme is used for collision avoidance [38]. A combination of visibility graphs and GNNs is used to achieve online collision avoidance. Conflict Detection and Resolution (CD&R) method is widely used for air traffic control. These methods predict the possibility of a conflict between two aircraft, and compute a manoeuvre strategy for the UAV such that conflict is avoided [39]. Ref. [78] presents a decision-making algorithm for pair-wise non-cooperative aircraft collision avoidance. More details of collision avoidance methods are discussed in Chapter 2.

1.5 Contributions of the thesis

The objective of this project is to develop a new advanced algorithm to support safety-critical OAS for unmanned vehicles. The main contributions of this thesis are:

- A novel optimisation-based safety analysis of OAS for unmanned vehicles is proposed and verified the obstacle avoidance algorithms in the presence of all possible parameters variations. This optimisation-based verification method can be applied to linear and nonlinear robustness analysis and also to different static and moving obstacle avoidance algorithms. Therefore, it is a very flexible and efficient method for the robustness analysis of collision avoidance system. The objective function is defined subject to the uncertain parameters set of lower and upper bounds in the time domain to find the worst case distance during the obstacle avoidance manoeuvre. First, potential field method is chosen for the static and moving obstacle avoidance path planning, and this method is verified with our proposed safety-critical algorithm. Then, decision making collision avoidance method is applied to the moving obstacles and verified the collision avoidance systems. This optimisation-based safety-critical algorithm may be applicable to verify other collision avoidance algorithms after appropriate modification.

- As a benchmark, firstly, simple unicycle robot is considered, and potential field method is chosen as path planning and collision avoidance algorithm in 2D space. Secondly, the moving OAS for Pioneer 3-DX Unicycle robot using with artificial potential field method is presented. Three types of uncertainties are considered in this study. Eight most significant uncertain parameters are chosen for the worst-case analysis. Finally, based on a 6 Degree of Freedom (6DoF) kinematic and dynamic model of UAV, the path planning and collision avoidance algorithms for fixed and moving obstacles using with potential field method are developed in 3D space. Decision making collision avoidance algorithm (3D geometry) method is studied and developed for UAVs.

- After that, the safety analysis is carried out using with proposed optimisation-based verification algorithm. Gradient based local optimisation (*fmincon*), two different evolutionary global optimisation techniques (Genetic Algorithm and GLOBAL algorithm), and a deterministic global optimisation algorithms (DIviding RECTangles) are applied to the OAS, and the reliability and efficiency of each of these methods are compared. It is shown that the obstacle avoidance algorithm and controller function correctly in the presence of parameters variations.

- Moreover, random number generation of Monte Carlo simulations are carried out, and the MCM results are compared with worst case scenarios obtained by optimisation-based verification method.

Chapter 2

Literature Review

The purpose of this literature review is to provide the necessary background to be able to understand the future work of this project. This chapter first begins with an introduction of V&V methods. Then the static obstacle avoidance algorithms including Roadmaps, Cell Decomposition, and Potential Field Method are described. In addition, some moving obstacle avoidance algorithms are also presented. The final section of this chapter provides some conclusions about the obstacle avoidance algorithms, and discusses the suitable methods to be used for this work.

2.1 Verification and Validation (V&V) Methods

V&V techniques have always been an essential part in the development of products, because they offer the only way to judge the success of a development project. V&V comprise a set of techniques used in engineering to evaluate the quality of the products. The conventional view is that verification is the process of checking whether the system meets the specified requirements of the users, while validation is the process of checking whether the system meets the actual requirements of the users [40]. Boehm [41] memorably characterised the difference as follows:

- ***Verification** is building the system right.*

  (Answering the question: “Are we building the system right?)

- ***Validation** is building the right system.*

  (Answering the question: “Are we building the right system?)

Therefore, the purpose of verification is to ensure that selected work products meet their specified requirements. The purpose of validation is to demonstrate that a product fulfils its intended use when placed in its intended environment [42].

V&V activities are important because they [42]:

- Ensure that requirements are met.
- Remove defects from the product throughout a project's life cycle.
- Reduce the cost of poor quality.
- Ensure the product fulfils its intended use when placed in its intended environment.
- Improve the quality of the process and the product.

The purpose of V&V means that it is to find errors in an early stage of the development process. Therefore, it is much less expensive to correct the errors than later on. In practice, several design activities are carried throughout the system development lifecycle. At the start of the system development lifecycle, the end users and developers have to identify the system's need and then translate them into a set of specifications. Within this process, a collection of functional and non-functional requirements are identified. Functional requirements specify what functions the system must perform, whereas the non-functional ones define how the system must behave, in which case they might impose constraints upon the systems behaviour such as performance, security, or reliability [43].

The collection of these requirements represents a highly iterative process that ends when the requirements reach a level of maturity sufficient in order to initiate the development phase. At the end of the V&V process, the results are inspected in order to make an official decision on whether to accept the system or not for a specific usage. This is known as certification (also known as accreditation), and it is commonly performed by certification authority [43]. The safety case is an important document used to support certification. It contains a set of arguments supported by analytical and experimental evidence concerning the safety of a design. In the United States, different government organizations are responsible for the certification of different products. For example, the FAA (Federal Aviation Authority) is in charge of the certification of aircraft. Specifically, the FAA software certification is based on the standard RTCA/DO-178B [46]. The standard provides information about all aspects of the software certification process including the following sections: software planning process, software development process, software verification process, and the certification process.

The V&V techniques mainly include testing, simulation, model checking, and theorem proving. Many engineering solutions are required to meet a very high level of reliability, security, and performance, especially in safety-critical areas. There are number of different methods and techniques for performing V&V. These can be divided into four groups: informal, static, dynamic and formal techniques [43, 45]:

Informal techniques: Informal techniques rely on human interpretation and subjectivity, without any underlying mathematical formalism. Informal techniques which include audit, face validation, turing test, walkthroughs, etc. More complete description can be found in [45].

Static techniques: Static techniques are concerned with accuracy on the basis of characteristics of the static model design and source code. It does not require machine execution of the model, but mental execution can be used. They aim at checking the structure of the model, the dataflow and control flow, the syntactical accuracy, and the consistency. Therefore, in order to provide a full V&V coverage, they have to be applied in conjunction with dynamic techniques defined in the next category. Some of the most common static techniques are control analysis, cause-effect graphing, data analysis, etc.

Dynamic Techniques: In contrast to the static techniques, dynamic ones are based on the machine execution of the model in order to evaluate its behaviour. They do not simply examine the output of an execution but also watch the model as it is being executed. Consequently, the insertion of additional code into the model is needed to collect or monitor the behaviour during its execution. Debugging, execution testing, functional testing, fault/failure testing, sensitivity analysis, statistical analysis, and visualization/animation are examples of dynamic techniques.

Formal techniques: These are based on formal mathematical reasoning and proofs. Among them are model checking and theorem proving. For example Lambda calculus, logical deduction, proof of correctness, etc.

2.2 Application of Analysis Techniques for Flight Control Law

In [1], five verification techniques were applied to the flight control law. These five advanced techniques are given below:

I. μ - analysis
II. v –gap analysis
III. Polynomial-based clearance
IV. A bifurcation and continuation method
V. Optimisation-based clearance

The aim of this design challenge was to describe how these advanced methods can be applied to the verification process of flight control law and to demonstrate this on the basis of a benchmark model.

The μ - analysis method is most suitable for analysing linear criteria in the frequency domain, i.e. the stability margin criterion and the eigen-value criterion. The v-gap can be used to measure the difference between a nominal system and perturbed system (including uncertainties). This method is also most suitable for linear model in the frequency domain [1; 51].

Another technique of Polynomial-based analysis method checks the robust stability of a dynamic system by looking at the uncertain coefficients of the characteristic polynomial. This method is not suitable for nonlinear criteria [1].

Bifurcation theory can be used to analysis a system of nonlinear differential equations by assessing its steady and non-steady equilibrium solutions as a function of its state and input variables. This means that for the control law clearance problem of an aircraft model with uncertainties, the influence of any parameter can be analysed. This method can be used for linear and nonlinear analysis. A weakness in this method is that it does not guarantee that the worst case being found. However, this method could be powerful if combined with the optimisation method [1].

The optimisation-based method for linear and nonlinear analysis was carried out in [1]. By using an optimisation algorithm, the worst case can be found as a function of uncertain model parameters. The main requirement is the availability of a parametric model describing the system's dynamics. Various local and global optimisation routines are in use, and care should be taken to select the best method for the particular problem. This method is very flexible and can handle linear and nonlinear criterion in the frequency and time domain.

Among these five techniques, the optimisation-based approaches showed the most potential for improving the flight control law clearance process, due to their versatility, relative mathematical simplicity, and applicability to nonlinear simulation models and clearance criteria.

2.3 Holonomic and Nonholonomic Constraints

Kinematic constraints are classically divided into two classes: holonomic constraints and nonholonomic constraints. i.e., a holonomic constraint can be expressed as an explicit function of position variables only. However, a nonholonomic constraint requires a differential relationship, such as the derivative of a position variable [47-50].

Let us imagine a system with n generalized coordinates $q= [q_1, \dots\dots, q_n]^T$. Then, holonomic constraints can be written as following form:

$$F(q) = F(q_1, \dots\dots, q_n) = 0 \tag{2.1}$$

Equation express dependence at least one of coordinates from the other and decreases number of the degrees of freedom of the system. If the equation (2.1) includes also the time derivatives of the generalized coordinates, then the equation is

$$F(q, \dot{q}) = F(q_1, \dots q_n, \dot{q}_1, \dots \dot{q}_n) = 0 \tag{2.2}$$

Constraints of this type are called nonholonomic constraints and the system is called nonholonomic system.

Figure 2.1: Wheel Mobile Robots [44]

Due to the presence of wheels, a Wheel Mobile Robot (WMR) (Fig 2.1) cannot move sideways. This is the rolling without slipping constraint, a special case of nonholonomic behaviour. In general, a nonholonomic mechanical system cannot move in arbitrary directions in its configuration space [91].

For an example of nonholonomy: The simplest model of a nonholonomic WMR is that of the unicycle which is shown in Fig 2.2.

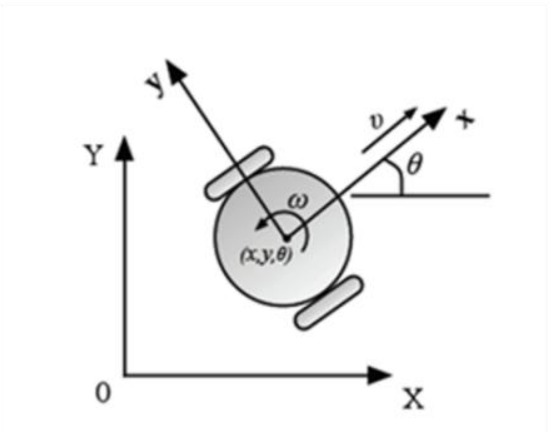


Figure 2.2: Unicycle mobile robot

The generalized coordinates are $q=[x, y, \theta]^T$. Kinematical state can be presented with its position in Cartesian system of coordinates x and y and its orientation θ:

$$\begin{aligned} \dot{x} &= v cos\theta \\ \dot{y} &= v sin\theta \end{aligned} \tag{2.3}$$

The pure rolling nonholonomic constraint is

$$\dot{x} sin\theta - \dot{y} cos\theta = 0 \tag{2.4}$$

2.4 Obstacle avoidance algorithms

Autonomous vehicles are widely used in many hazardous industrial fields such as aerospace research, the nuclear industry, military operations, etc. To find a safe path in a dangerous environment for the autonomous vehicles is an essential requirement for the success of any autonomous systems. Path planning algorithms to make the robot move from the start point to the goal point without collision with obstacles is a fundamental requirement for the mobile robot safety in such environments. Furthermore, the planned path is required to reduce the processing time, communication delay and energy consumption.

Autonomous vehicles obstacle avoidance approaches can be classified into global path planning and local collision avoidance algorithms [55]. Global path planning requires the totally known environment and the obstacles should be fixed (like buildings, trees, etc). In this path planning, the robot or UAV is the only one that moves. The global path planning algorithms deals with finding a suitable path from an initial point to a target point using a given representation of the environment. Search algorithms are used to find the suitable path in this approach because the entire environment is known. Therefore, the suitable path for the vehicle could be the global optimised result, and these methods are usually computationally expensive. This global path planning methods are not suitable for avoiding collisions with pop-up and moving obstacles and cannot be executed online. On the other hand, local path planning means that path planning is done in a partial known or unknown environment. This algorithm is capable of producing a new path in response to environmental changes, i.e. an onboard sensor detects the possibility of collision and then an alternative route is planned online. For example, assume that there are no obstacles from the start point to goal point. The robot will move along this straight line path until an obstacle is detected. Local path planning algorithms are able to react quickly to change in the environment. This algorithm is applied to find the alternative path around the obstacles. Therefore, local path planning algorithms can only develop plans for the immediate future.

Multi-layer architectures have been implemented in recent collision avoidance approaches. A global planning layer first finds a dynamically feasible obstacle-free optimal path to the goal, and then a local collision avoidance layer reacts to changes in the environment and computes an alternate, collision-free path online [53]. In [54], a multi-layer approach for motion planning in obstacle rich environments is built on the principle of separation of concerns which partitions a given problem into multiple independent layers. Computational geometry layer and an optimal layer are used in this motion planning algorithm.

Literature survey of static and moving obstacle avoidance algorithms is presented in this chapter.

2.5 Static obstacle avoidance algorithms

The path planning problem is formulated in the configuration space representation. A configuration is a complete specification of the position of every point in the system. The configuration space (c-space) is the space of all possible configurations of the robot (See Figs. 2.3-2.4). Thus, each pose (position and orientation) of a robot is represented by a point in the c-space while the obstacles are expanded appropriately. Then, the path-planning problem becomes equivalent to the path planning of a point robot in c-space [19]

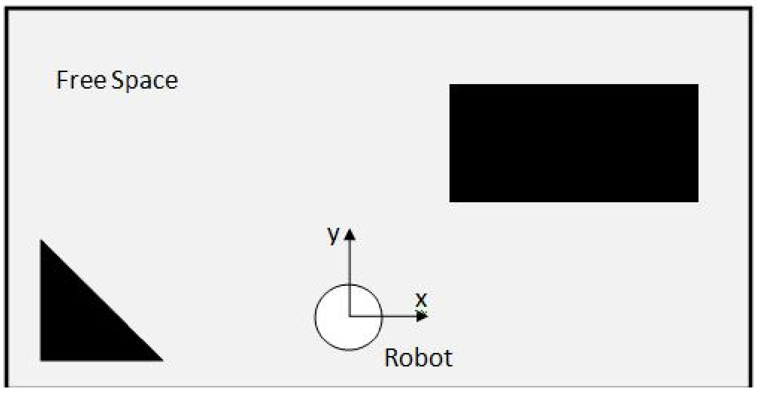


Figure 2.3: Physical space

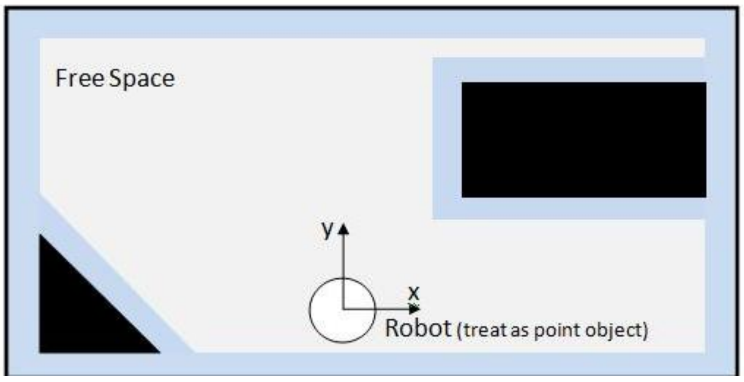


Figure 2.4: Configuration Space

The most three common approaches for static obstacle avoidance are the roadmap, cell decomposition, and potential field method. Roadmaps model connections between special points, cell decomposition methods break the world into grids, and potential field methods apply mathematical fields to model the world. Roadmap and cell decomposition methods must be discrete, and potential field method can be implemented in a continuous state space [55]. More details of these methods are discussed below:

2.5.1 Roadmaps

The Roadmap is graphs which represent how to get from one place to another. Using a roadmap, the planner can construct a path between any two points in a connected component of the robot's free space by first finding a collision-free path onto the roadmap, traversing the roadmap to the vicinity of the goal, and then constructing a collision-free path from a point on the roadmap to the goal [19]. The nodes in the graph are usually waypoints that the robot needs to travel between for a successful journey.

Roadmaps provide a huge advantage over cell decompositions in the number of nodes, and the planner needs to search through in order to find a path. The set of nodes does not consist of all of the configurations, but a select few that are special. This makes them harder to create, but easier to manipulate and use. On the other hand, roadmaps are generally difficult to update or repair as the robot gains new information, because the entire roadmap typically needs to be remade [55].

There are several ways that such a connectivity map can be built up. The well-known approaches are Visibility graph, Voronoi diagram, and probabilistic roadmaps. The most influential sampling-based algorithms are Probabilistic RoadMaps and Rapidly-Exploring Random Trees (RRT) [19; 20].

a. **Visibility graph**

In this approach, the obstacles are usually polygons and the set of possible paths are the straight line segments. The main idea of the visibility graph method is that if there is a collision-free path between two points, then there is a polygonal path that bends only at the obstacle vertices. As Fig 2.5 shows that collision-free path (in curves) could be transformed into line segments (straight line) [19].

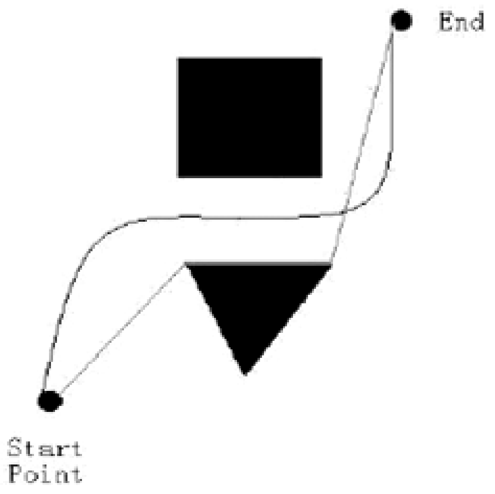


Figure 2.5: Visibility graph

A visibility graph is constituted by nodes and edges. Nodes are the start point, destination point and the vertices of all obstacles. Edges are straight-line segment between two nodes which do not path through obstacles. An example of visibility graph is shown in Fig 2.6. In this example, the possible paths which can be taken by the roadmap are shown in solid lines connecting the corners of the obstacles, and the shortest path through the roadmap, which the robot would take, is shown as the dotted line [50 ; 55]. Fig 2.7 shows that are multiplex paths that could lead the robot from the start point to the destination. Then, any search algorithms such as genetic algorithms, simulated annealing algorithm, A^*, Dijkstra, etc could be used to calculate the optimal path for the robot. There are some disadvantages in this method. The efficiency of the algorithm is low. In addition, the obtained path is often very close to obstacles and, thus may lead to crashing of the robot. However, this problem can be fixed by enlarging the obstacles by a value according to the dimension of the robot. In this way, the robot can approaches obstacles without collision [19]. Another problem is that the obstacles must be clearly defined polygons. This is a problem for outdoor robots because obstacles almost always take on round or amorphous shapes. This method is more suitable for static obstacle avoidance.

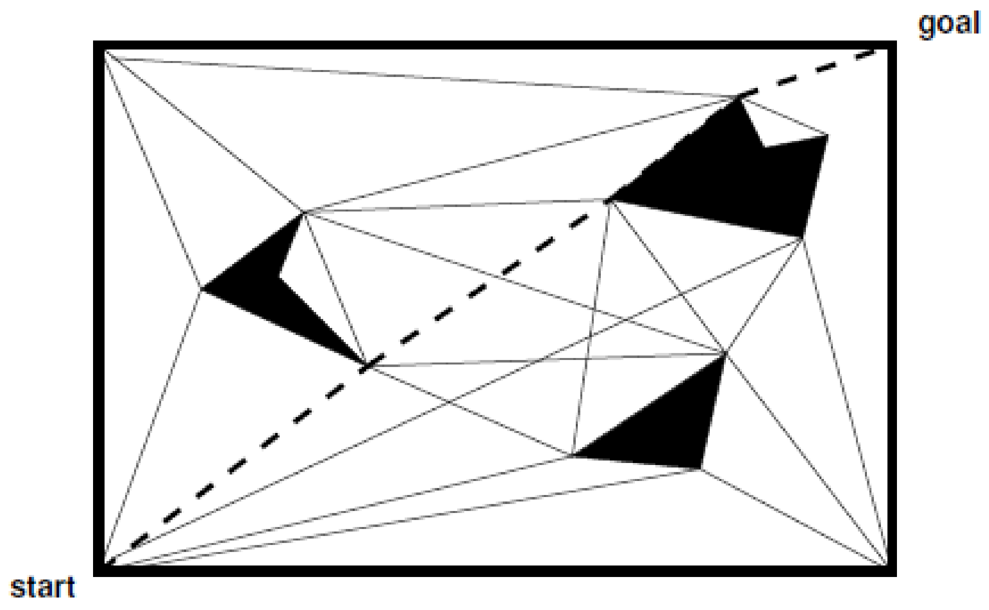


Figure 2.6: Multiplex path- Visibility graph

b. Voronoi diagram:

It is another popular method for generating a roadmap from c-space. It can be constructed as the robot enters a new environment. The roadmap consists of paths, or Voronoi edges, which are equidistant from all the points in the obstacle region. A Voronoi diagram is shown in Fig 2.7 [55]. The points where these edges meet are called vertices. McKerrow [57] stated that Voronoi diagram can be used to divide the environment into regions. Lee and Drysdale [58] showed how partitioning the plane into polygonal regions, each of which is associated with a given point, forms a Voronoi diagram.

In this planner, firstly, the environment is set up. The Voronoi diagram is then constructed based on the points of the obstacles and the boundary. The diagram is then pruned so that only the lines outside of the obstacles remain. The start and final configurations are then connected to the pruned diagram. Finally, the lines are smoothed and a search algorithm (Dijkstra) is performed to find the shortest path. In contrast to visibility graphs, Voronoi paths are as far as possible from the obstacles. Therefore, there is no need to grow obstacle boundaries [19; 55]. This approach is a very attractive method to use in low dimensions. It is accurate, fast and produces a desirable path. However, most Voronoi-based methods have the difficulty of calculating the Voronoi Diagram by studying lines and polygons, finding the vertices and nodes, and creating a tree to find the path. In addition, the method is not very efficient in three dimensions and it does not work at all in higher dimensions [56].

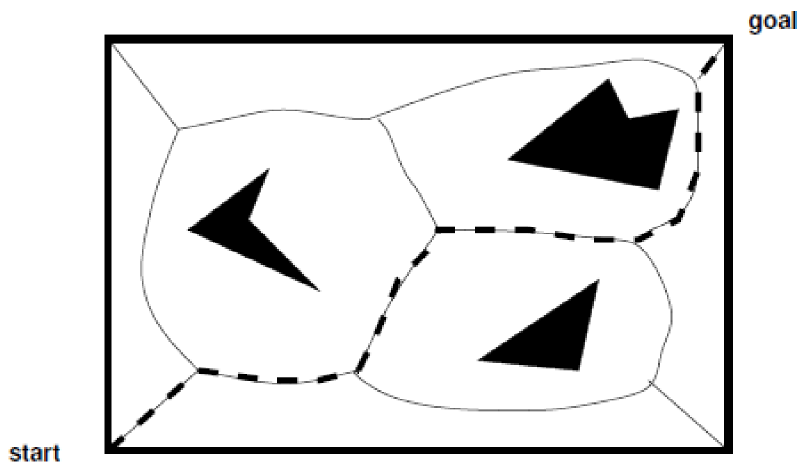


Figure 2.7: A Voronoi diagram

c. Probabilistic RoadMap (PRM) Planners:

PRM planning is one of the most efficient methods to compute collision free paths for vehicles. It is more suitable for robots with many degrees of freedom. This method consists of two phases: a building phase and a query phase [19, 59].

In the building phase a roadmap is built. The roadmap consists of nodes with collision-free configurations and edges corresponding to collision-free paths between adjacent nodes. The roadmap is constructed by repeating the two following steps. Firstly, pick a random collision-free configuration of the robot, and then connect the configurations by using a simple and fast planner, called local planner. In the query phase, a search for a path between an initial and a goal configuration is performed. Firstly, a path from the start and the final configurations to two nearby nodes in the roadmap is found. Then, a graph search in the roadmap is performed, resulting in a sequence of edges connecting these two nodes. This method was originally developed for nonholonomic robots in a static environment [19, 59]. A simple example of a PRM and a path from start to goal is shown in Fig 2.8.

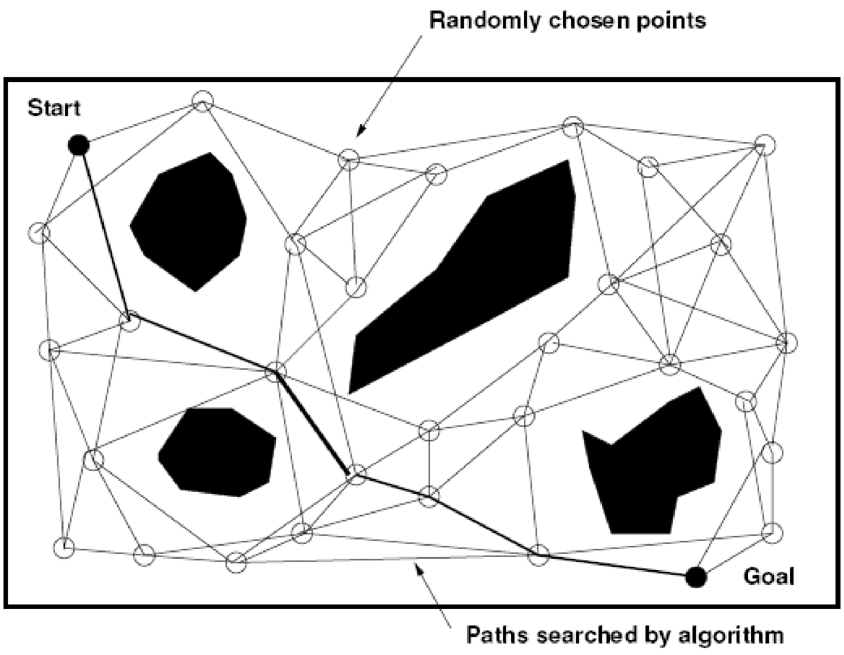


Figure 2.8: PRM with randomly chosen nodes

One of the advantages of the PRM algorithm is that the initial graph building process is computationally expensive, however, once it has been constructed, the search is very efficient. The problem with PRM is that when obstacles are added or removed from the map and the entire roadmap must be regenerated. In addition, another problem is that obstacles need to be well defined and generating variable path costs is more difficult than with other methods [19; 55].

d. Rapidly Exploring Random Trees:

A further variation of PRMs is the Rapidly Exploring Random Tree (RRT). Rather than randomly sampling the c-space as a PRM does the planner begins at the start location and randomly expands a path, or tree, to cover the c-space [55]. Lavalle [60] introduced the RRT as a planning approach to quickly search high-dimensional spaces with both algebraic constraints (arising from obstacles) and differential constraints, which can be applied to a wide variety of planning problems with nonholonomic constraints. One method for planning as shown in Fig 2.9 is to grow two RRTs, one from the goal and one from the start, and then search for states that are common to both, creating a linked path between the two.

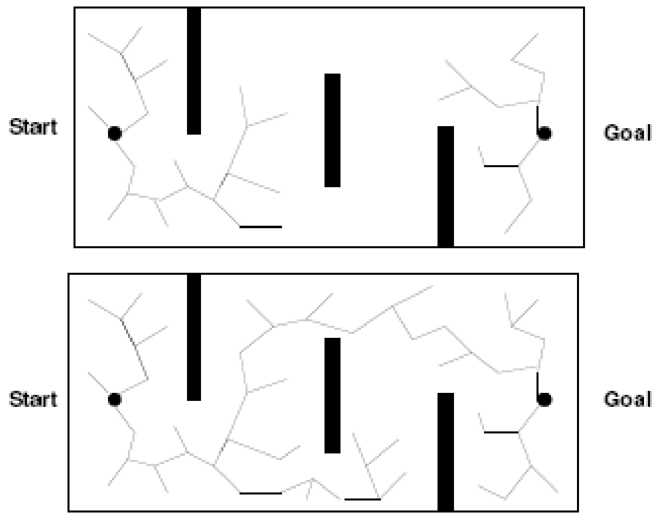


Figure 2.9: Path planning using standard RRTs

An important disadvantage of RRT is that avoiding collision with moving obstacles may not be possible, because of the random nature of the algorithm. Additionally, it is an open-loop method, and thus is sensitive to modelling inaccuracies and external noise such as wind. In order to avoid this problem, Kuwata et al [61] proposed the overall closed-loop CL-RRT framework. The main challenges in designing the motion planning subsystem resulted from the following factors [61]: complex and unstable vehicle dynamics, with substantial drift; limited sensing capabilities, such as range and visibility, in an uncertain, time-varying environment; and temporal and logical constraints on the vehicle's behaviour, arising from the rules of the road. This CL-RRT approach has several advantages when compared to the standard approach. First, CL-RRT works for vehicles with unstable dynamics, such as cars and helicopters, by using a stabilizing controller. Second, the use of a stabilizing controller provides smaller prediction error because it reduces the effect of any modelling errors in the vehicle dynamics on the prediction accuracy, and also rejects disturbances/noises that act on the actual vehicle. Third, the forward simulation can handle any nonlinear vehicle model and/or controller. Finally, a single input to the closed-loop system can create a long trajectory while the controller provides a high-rate stabilizing feedback to the vehicle. This requires far fewer samples to build a tree, improving the efficiency of randomized planning approaches.

2.5.2 Cell Decomposition

Another class of global planning methods is Cell Decomposition (CD). The idea behind CD is to decompose the c-space into a number of disjoint sets, called cells. An important element of CD methods is the connectivity graph *G* that captures the structure of c-space. Each cell is represented as a node in this graph. Two nodes are connected by an edge if and only if the two corresponding cells are adjacent [63]. From this graph a continuous path or channel can be determined by simply following adjacent free cells from the start point to the goal point.
CD methods can be classified as exact or approximate. The major difference is that exact CD results in cells of different simple shapes as required by the shape of obstacles. Approximate CD methods use predetermined cell shapes, sizes, and positions to approximate the free space [62].

a. Exact Cell Decomposition

Exact CD attempts to solve some of the problems with regular grids in a different way. The cells do not have a predefined size or shape, but are determined based on the world map and the location and shape of obstacles within it [55]. An exact CD using rectangular strips with 3 polygonal obstacles is shown in Fig.2.10. The cells joining the start and target are shaded. Let us consider robot motion planning reduced to navigating a point in a free space F. Then the CD can be stated as follows [66]:

1. Divide F into connected regions called cells.
2. Determine which cells are adjacent and construct an adjacency graph. The vertices of this graph are cells, and edges join cells that have a common boundary.
3. Determine which cells the start and goal lie in, and search for a path in the adjacency graph between these cells.
4. From the sequence of cells found in the last step, compute a path connecting certain points of cells such as their midpoints via the midpoints of the boundaries.

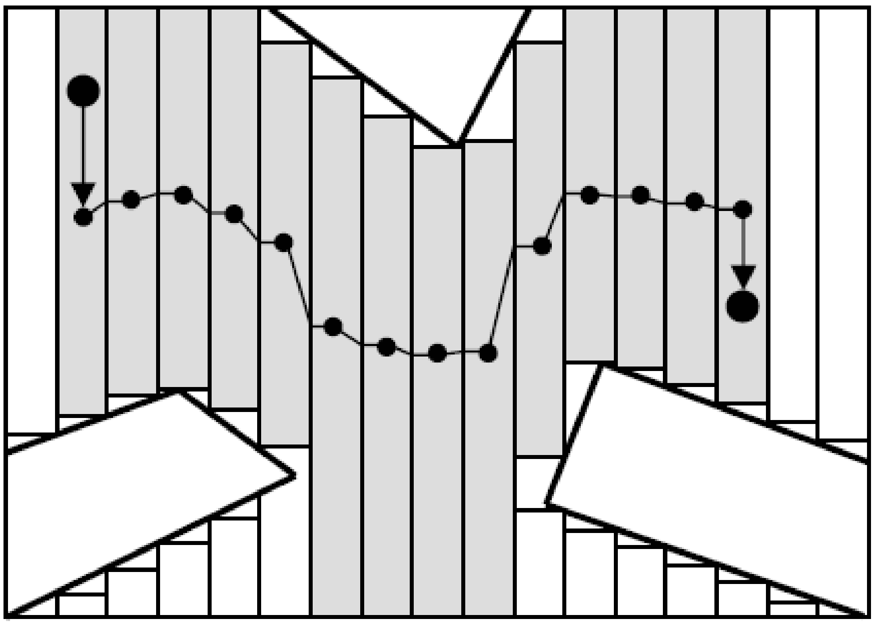


Figure 2.10: Exact Cell Decomposition

b. Approximate Cell Decomposition

Approximate CD is created by laying a regular grid over the planning space. The cells of the grid are of a predefined shape and size, and are therefore easy to apply. If there is an object in the area contained by the grid element, that element is marked as an obstacle. Otherwise it is left as free space. The center of each cell becomes a node in the search graph that will be examined to find a path. These nodes can either be 4-connected or 8-connected representing whether or not the robot is considered to travel diagonally between them (See Fig 2.11). Approximate CD is popular for a number of reasons: the algorithms are easier to implement, they are simple to apply to a world space, and they are flexible. However, there are a few drawbacks in this method. In CD methods an obstacle much smaller than the grid size will result in that entire grid square being labelled as occupied. This results in a conservative estimate of the free space. Also, the complexity of these methods grows quickly with the dimension of the c-space so they are realistically applicable only when the c-space has dimensions of around 4 or less [55].

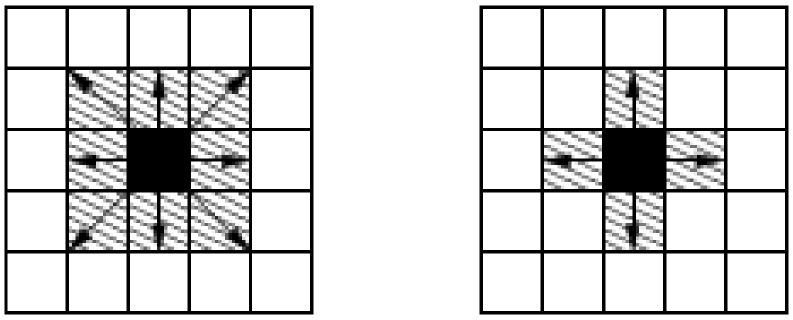


Figure 2.11: 8-connected and 4-connected grids

2.5.3 Potential Field Method

An another approach is the Potential Field Method, an idea of adding an imaginary forces on the robot, was first suggested by Khatib [64] in early 80’s for obstacle avoidance of manipulators and mobile robots. Potential field methods are quite different from the previously discussed methods of planning, and have been used extensively in the past. Instead of trying to map the search space they impose a mathematical function over the entire area of robot travel. In its simplest form, potential field method can be implemented quickly and provide acceptable results without requiring many calculations. In this method, the obstacle exerts a repulse force on the robot, while the goal position applies attractive force to the robot. The sum of the attractive and repulse forces is then used to determine the direction and the speed of the robot [19]. Such approaches have been variously termed as Potential Field Approaches, Artificial Potential Methods, Virtual Potential Approaches, Potential Based Approaches, or Potential Field-based Navigation Methods [52].

The potential function approach directs a robot as if it were a particle moving in a gradient vector field. Gradients can be intuitively viewed as forces acting on a positively charged particle robot which is attracted to the negatively charged goal. Obstacles also have a positive charge which forms a repulsive force directing the robot away from the obstacles. The combination of attractive and repulsive forces hopefully directs the robot from the start location to the goal location while avoiding obstacles. Variants of the attractive and repulsive functions are described below:

a. Attractive potential field

There are many different attractive potential function have been proposed in the literature, the most commonly used attractive potential field takes the form of

$$U_{att}(q) = \frac{1}{2} k_a \left\| q - q_{goal} \right\|^m \tag{2.5}$$

where $U_{att}(q)$ denotes the attractive potential field, k_a is a positive scaling factor, $q=[x, y]^T$ is the position for the robot in 2D workspace and $q=[x, y, z]^T$ in a 3D, q_{goal} denotes the position for the goal, $\left\| q - q_{goal} \right\|$ is the Euclidean distance between the robot and the goal. m is any positive number greater than zero [19, 52].

We could easily get the gradient information by:

$$\nabla U_{att}(q) = \frac{\partial U_{att}(q)}{\partial q} \tag{2.6}$$

The negative gradient of the attractive potential is considered as the attractive force. While the gradient $\nabla U_{att}(q)$ converges linearly to zero as q approaches q_{goal}, it grows without bounds a q moves away from q_{goal}. If q_{start} is far from q_{goal} , this may produce a desired velocity that is too large. For this reason, combine the quadratic and conic attractive potential may be chosen so that the conic potential attracts the robot when it is very distant from q_{goal} and the quadratic potential attracts the robot when it is near q_{goal}. So, the gradient is defined at the boundary between the conic and quadratic portions. Such a field can be defined by

$$U_{att}(q) = \begin{cases} k_a\|q - q_{goal}\|^2 & , \quad \|q - q_{goal}\| \le s \\ 2ks\|q - q_{goal}\| - ks^2 & , \quad \|q - q_{goal}\| > s \end{cases} \tag{2.7}$$

where s is the threshold distance from the goal where the planner switches between conic and quadratic potential [19]. In Fig 2.12, an attractive potential field is created using equation (2.7). The goal is positioned at (6, 7). k_a and s are chosen as 5 and 1 respectively.

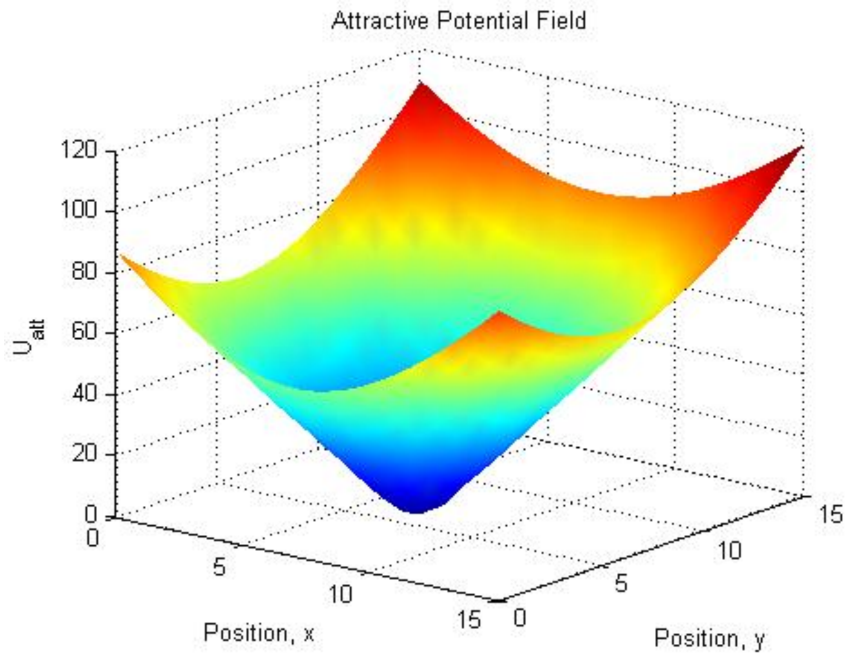


Figure 2.12: Attractive Potential Field

b. Repulsive potential field

A repulsive potential keeps the robot away from an obstacle. The closer the robot is to an obstacle, the repulsive force should be stronger. Therefore, the repulsive potential is usually defined in terms of distance to the closest obstacle d_{obst}. Different types of repulsive potential were proposed in the literatures. The most widely use repulsive potential functions are discussed below:

The repulsive potential field of the Force Involving and Artificial Repulsion from the Surface Function (FIRAS function) was proposed by Khatib [64]. The FIRAS functions have spherical symmetry i.e. modelled the obstacle as a circular disc. Therefore, this FIRAS function is not suitable for long shape obstacles. The FIRAS function is described by:

$$U_{rep}(q) = \begin{cases} \frac{1}{2} k_r \left(\frac{1}{d_{obst}} - \frac{1}{d_0} \right)^2, & \text{if } d_{obst} \leq d_0 \\ 0 \qquad\qquad\qquad\quad , & \text{if } d_{obst} > d_0 \end{cases} \tag{2.8}$$

where $U_{rep}(q)$ is the repulsive potential field, k_r is a positive scaling factor, d_{obst} is the shortest distance between the robot from the obstacle, and d_0 is the limit distance of the repulsive potential field influence.

In [67, 68], superquadric repulsive potential function was used to solve the problem of local minima. However, the superquadric potential function can only guarantee global minima with single obstacle. It cannot solve the problem of local minima resulted from multiple obstacles, such as when obstacles arranged in a U shape. Kim and Khosla [69] proposed a harmonic potential function for the obstacle avoidance problem. Harmonic potential function was used to eliminate the local minima problem. However, it will create a structural local minimum. The structural local minimum is a static equilibrium where the robot cannot move any more with a current artificial potential. This is the one type of local minimum that results from multiple obstacles. And also, the simplest form of this potential field can only model point obstacles.

After that, Ge and Cui [72] proposed a new potential function which is called as GNRON (Goal Non-Reachable with Obstacles Nearby). It was modified the FIRAS function intended to solve the problem (i.e. Goal Non-Reachable with Obstacles Nearby) found in FIRAS function.

This function takes the following form:

$$U_{rep}(q) = \begin{cases} \frac{1}{2} k_r \left(\frac{1}{d_{obst}} - \frac{1}{d_0} \right)^2 d_{goal}^n, & \text{if } d_{obst} \leq d_0 \\ 0 & , \text{ if } d_{obst} > d_0 \end{cases} \tag{2.9}$$

where d_{goal}^n is the minimal Euclidean distance from robot to the target. Compare GNRON equation with the FIRAS function, the introduction of the term d_{goal}^n ensures that the total potential will reach its global minimum, if and only if the robot reaches the target where $d_{goal}^n = 0$.

c. Total Potential Field

Therefore, the total potential field is obtained by adding the repulsive potential resulting from all obstacles and the attractive potential from the target as given below:

$$U_{Total}(q) = U_{att(q)} + U_{rep}(q) \tag{2.10}$$

where $U_{Total}(q)$ denotes the total artificial potential field.

There are two most important disadvantages in potential field method.

Trap situation: Encountering traps, which are local minima in the potential function due to the arrangement of the obstacles, is one of the most common problems with potential field. This situation may occur when the vehicle runs into a dead end such as a U-shaped obstacle [19, 59]. Fig 2.13 illustrates this problem, where *R* represents the resultant force of the potential field.

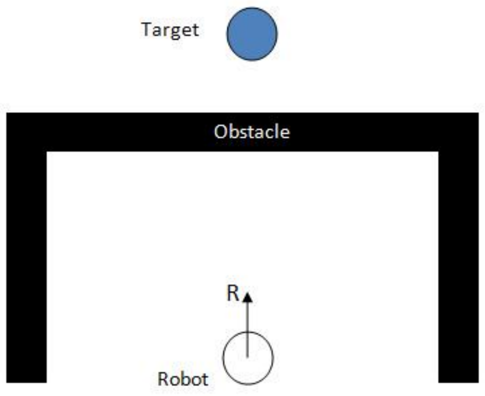


Figure 2.13: Trap situation due to local minima

No passage between closely spaced obstacles: This situation arises when a vehicle tries to pass between two closely spaced radars. The combined force of repulsion of both radars along with the attractive force of the goal location will prevent the vehicle from using a path that in realty it should be able to take. In Fig 2.14, F_r is the resultant repulsive force from the two radars and F_a is the attractive force pulling the robot towards the goal location. The vector sum of these two forces is R and this will be the vector that the robot will follow. Although the vehicle could physically fit between the two obstacles, the potential field approach does not generate solution [19, 59].

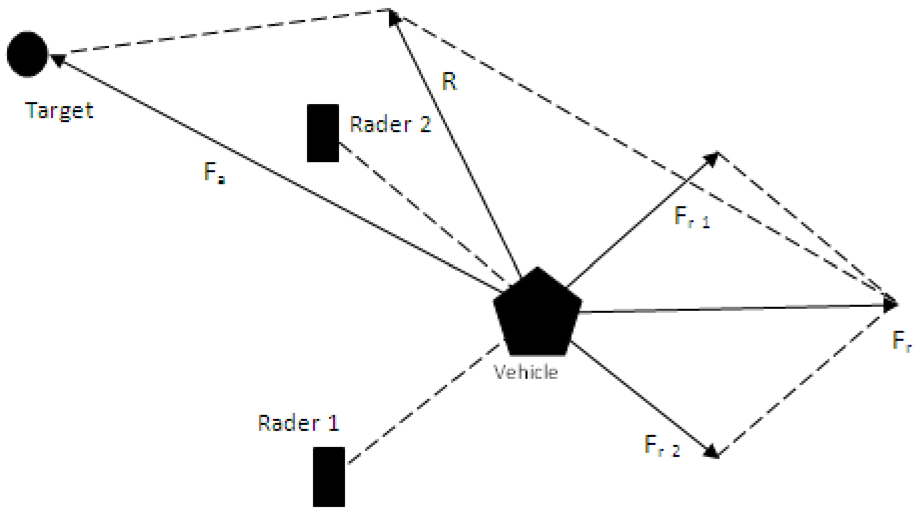


Figure 2.14: No passage between closely spaced obstacles

2.6 Moving Obstacle Avoidance Algorithm

2.6.1 Potential Field Method

Ge and Cui [74] proposed a potential field method for motion planning of mobile robots in a dynamic environment where the target and the obstacles are moving. The attractive potential is defined as a function of the relative position and velocity of the target with respect to the robot. The repulsive potential is also defined as the relative position and velocity of the robot with respect to the obstacles. The virtual force is defined as the negative gradient of the potential in terms of position and velocity rather than position only [74]. The attractive potential field used by Ge and Cui [74] has the following form:

$$U_{att}(p,v) = \alpha_p \left\| p_{goal}(t) - p(t) \right\|^m + \alpha_v \left\| v_{goal}(t) - v(t) \right\|^n \tag{2.11}$$

where v_{goal} and *v* denote the velocities of the goal and robot at time *t* respectively. $\mathrm{p}_{goal}(t)$ and *p(t)* denote the positions of the robot and the target at time *t*, respectively. $\left\| v_{goal}(t) - v(t) \right\|$ is the magnitude of the relative velocity between the goal and the robot at time *t*. $\| p_{goal}(t) - p(t)$ is the Euclidean distance between the robot and the relative velocity between the target and the robot at time *t*. α_p and α_v are positive scalar parameters. *m* and *n* are positive constants.

This attractive potential field has advantage that if we have a moving target: the first term in Eq (2.11) drives the robot to the target and shortens the distance between them while the second term drives the robot to move at the same velocity of the target. Therefore, the attractive force is defined as

$$F_{att}(p,v) = -\nabla_p U_{att}(p,v) - \nabla_v U_{att}(p,v) \tag{2.12}$$

where

$$\nabla_p U_{att}(p,v) = \frac{\partial U_{att}(p,v)}{\partial p} \tag{2.13}$$

$$\nabla_v U_{att}(p,v) = \frac{\partial U_{att}(p,v)}{\partial v} \tag{2.14}$$

To solve moving obstacle avoidance, repulsive potential field is defined in the terms of relative positions and velocities between the robot and the obstacles. The obstacles are convex polygons whose shapes, positions p_{obs} and velocities v_{obs} can be measured on-line. $i = 1,2, \ldots\ldots, n_{obs}$; where n_{obs} is the number of obstacles. The repulsive potential field is defined as follows:

$$U_{rep}(p,v) = \begin{cases} 0, \quad \text{if} \quad \rho_s(p,p_{obs}) - \rho_m(v_{RO}) \geq \rho_0 \ \text{ or } \ v_{RO} \leq 0 \\ \eta\left(\frac{1}{\rho_s(p,p_{obs}) - \rho_m(v_{RO})} - \frac{1}{\rho_0}\right), \quad \text{if} \ \ 0 < \rho_s(p,p_{obs}) - \rho_m(v_{RO}) < \rho_0 \ \text{ and } \ v_{RO} > 0 \\ \text{not defined} \ \ , \ \text{if} \ \ v_{RO} > 0 \ \text{ and } \ \rho_s(p,p_{obs}) < \rho_m(v_{RO}) \end{cases} \tag{2.15}$$

where ρ_0 is a positive constant describing the influence range of the obstacle; η is a positive constant; v_{RO} is a relative velocity between the robot and the obstacle in the direction from the robot to the obstacle; ρ_s is the shortest distance between the robot and the body of the obstacle; ρ_m is the distance travelled by the robot before v_{RO} reduces to zero. Similar to the definition of the attractive force, the corresponding repulsive force is defined as the negative gradient of the repulsive potential in terms of both position and velocity

$$F_{rep}(p,v) = -\nabla_p U_{rep}(p,v) - \nabla_v U_{rep}(p,v) \tag{2.16}$$

The total virtual force F_{Total} is the combination of attractive force and repulsive force. The total virtual force is used for motion planning. More details can be found in [74].

2.6.2 Conflict Detection and Resolution

A conflict detection and resolution (CD&R) are used widely for air traffic control. CD&R is described by using simple geometric approach. ‘Conflict’ can be defined as a “predicted violation of a separation assurance standard” [75]. So, if the protected zone is violated, each UAV should solve the violation using proper way to avoid the conflict.

The miss distance vector of two UAVs and the time to take are calculated using with PCA (Point of Closest Approach method [76 ; 77]). If the magnitude of miss distance vector is smaller than the minimum separation which should be guaranteed, it is considered as a conflict that can bring about collision between UAVs. Two UAVs are dealt with and considered as point masses with constant velocities toward their goal positions. Initially the positions and velocities are assumed to be informed by certain broadcasting systems, and the information from such as GPS is assumed to be quite exact. And also, it is assumed that two UAVs are in encounter with each other as shown in Fig 2.15, and they are heading in their velocity direction, i.e. there is no sideslip. The $\overrightarrow{r_m}$ is the miss distance vector which is found from the PCA method is defined as

$$r_m = \hat{c} \times (\vec{r} \times \hat{c}) \tag{2.17}$$

where $\vec{r}$ is the relative distance vector; $\hat{c}$ is the unit vector in the direction of the relative velocity $\vec{c}$ from UAV 'A' to UAV 'B' [77].

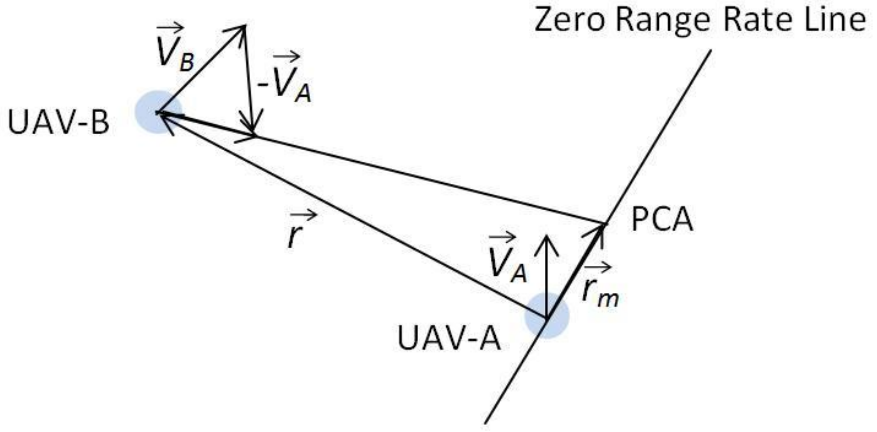


Figure 2.15: Relative motion of two UAVs

The time of closest approach is found to be

$$\tau = -\frac{\vec{r}.\vec{c}}{\vec{c}.\vec{c}} \tag{2.18}$$

It is found that when the time of approach $\tau > 0$ there is a chance of conflict. There is no chance of collision when $\tau < 0$. Therefore, the safety distance has to be checked when $\tau > 0$. If the magnitude of $\overrightarrow{r_m}$ is less than specified minimum separation distance r_{sa} ($r_m < r_{safe}$), there is a conflict, which must be resolved.

For conflict resolution, a resolution manoeuvre must be computed, which lies along the miss distance vector as shown in Fig 2.16. $\overrightarrow{V_A}$ and $\overrightarrow{V_B}$ are the actual velocity directions of UAV 'A' and UAV 'B' respectively. $\overrightarrow{U_A}$ and $\overrightarrow{U_B}$ are the velocity directions the UAV must go along so that the distance between the UAVs r_{safe} , r_{VSA} and r_{VSB} are the vectors of each UAV along the miss distance vector such that

$$|r_{safe}| = |r_{VSA}| + |r_{VSB}| + |r_m| \tag{2.20}$$

The main disadvantage of this method is that communication links used at the present time to relay information among UAVs cannot guarantee perfect information transfer. In addition, the algorithm is developed for an obstacle moving with constant velocity.

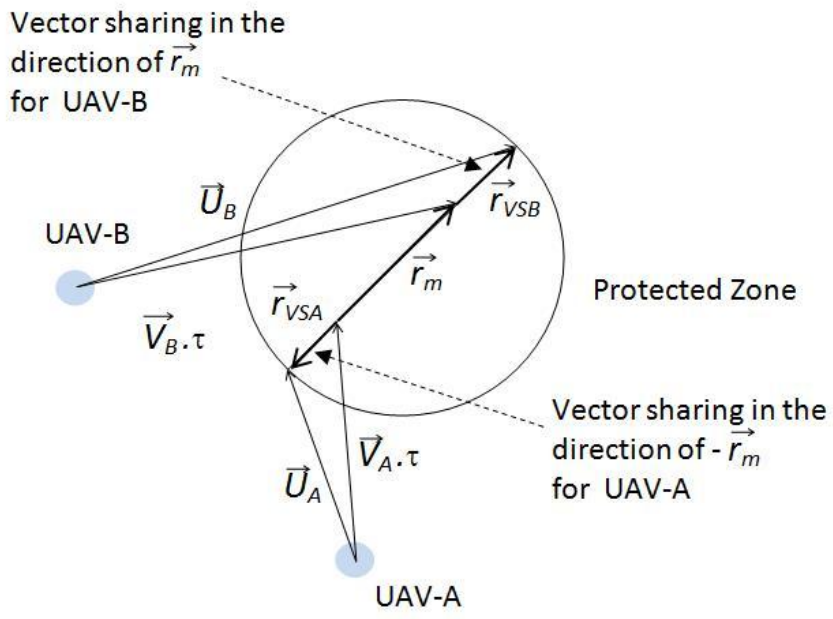


Figure 2.16: Vector sharing resolution to resolve the conflict

2.6.3 Decision-Making Collision Avoidance Algorithm

In [78], a fully autonomous multi-sensor anti-collision system for UAVs is presented. Autonomous Collision Avoidance (ACA) decision-making can be formulated as a two stage process. Firstly, Conflict Detection, a potential conflict between two aircrafts will be detected, determining if the future positions, after a certain amount of time should experience a loss of minimum separation. In such a case the trajectory of aircraft A/C_A has to be re-planned by solving a Conflict Resolution problem [78 ; 79].

The aircraft with ACA module on-board A/C_A modelled as a point object with 3 degree of freedom and velocity $\overrightarrow{V_A}$, while the other aircraft (A/C_B, considered as an intruder) is modelled as a sphere with radius R (safety bubble) having velocity $\overrightarrow{V_B}$. Fig 2.17 outlines the ACA module within the closed-loop control system. Its core is represented by a decision-making algorithm, having as an input the speed and the position of the intruder $(\overrightarrow{V_B}, \overrightarrow{P_B})$ and of the own aircraft $(\overrightarrow{V_A}, \overrightarrow{P_A})$ (See Fig. 2.18). The outputs of the decision making algorithm are reference signal to the autopilot, in terms of demanded speed module (V_d), slope angle (γ_d) and track angle χ_d.

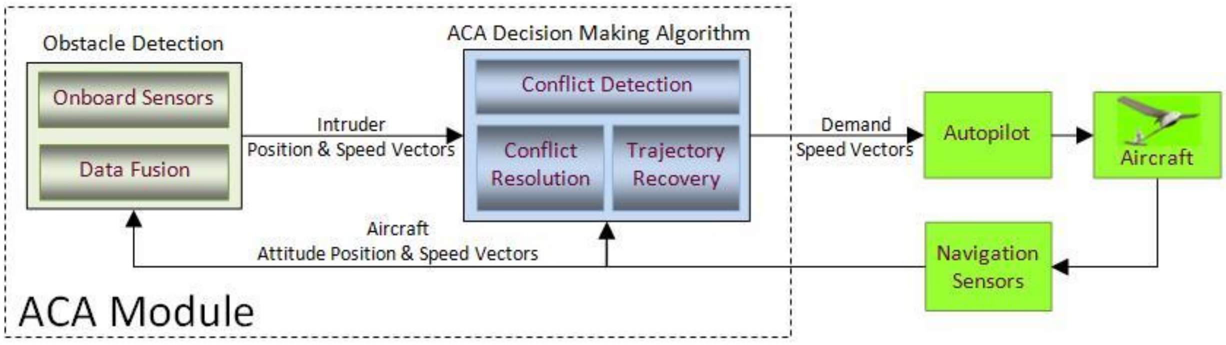


Figure 2.17: Autonomous collision avoidance control systems

A relative velocity vector $\overrightarrow{V_{AB}} = \overrightarrow{V_A} - \overrightarrow{V_B}$ transforms a dynamic collision avoidance problem into a static problem. Let $\overrightarrow{d_{AB}}$ be a vector defined as the minimum separation distance experienced between aircraft, after a certain time horizon. It can be calculated as follows:

$$\overrightarrow{d_{AB}} = \frac{\vec{r}.\overrightarrow{V_{AB}}}{\left\|\overrightarrow{V_{AB}}\right\|^2}\vec{V}_{AB} - \vec{r} \tag{2.21}$$

If a point and a circle are moving with constant velocities such that their initial conditions satisfy

$$\left\|\vec{d}_{AB}\right\| \leq R \text{ and } \dot{r} < 0 \tag{2.22}$$

Then they are headed for a collision. The above conditions are both necessary and sufficient for a collision to occur.

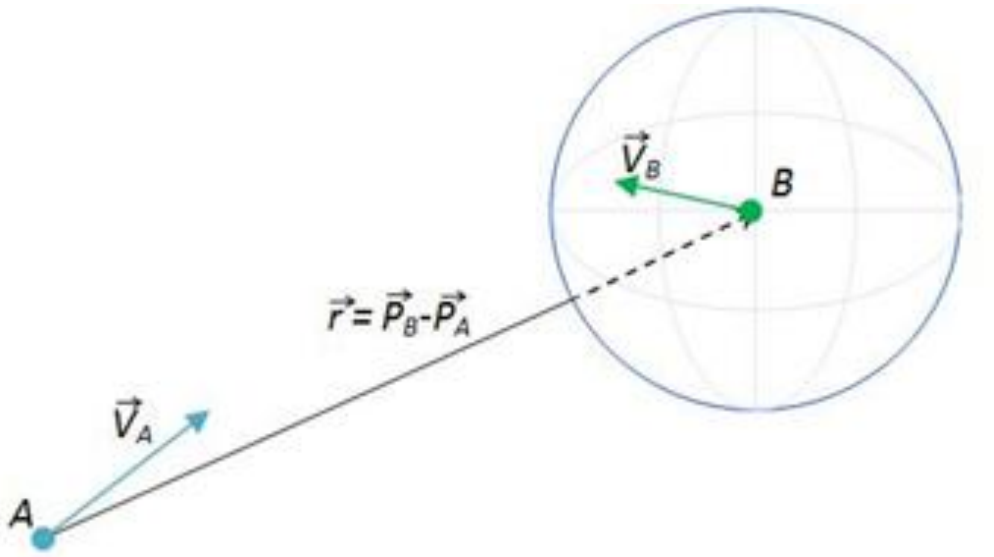


Figure 2.18: Distance between a point of mass A and a sphere B

2.7 Conclusion

The background of the collision avoidance algorithms are presented in this chapter. Researchers distinguish between various methods used to solve the obstacle avoidance problem according to two factors, global path planning algorithms and local obstacle avoidance algorithms which are presented in this chapter.

Potential field methods have also been extensively used for obstacle avoidance for single mobile robots, multiple mobile robots and moving obstacles. As a summary, a various types of potential field methods have been studied for finding a path for the mobile robot. Nowadays, the artificial potential field method is combined with many other computational methods to improve its efficiency. In [80], the survey reveals that the potential field method has been applied to various robot motion planning in the last three decades. Therefore, the artificial potential field method is chosen as a path planning and obstacle avoidance candidate technique for verification study as it is simple and widely used. However, the verification technique proposed in this thesis may be applicable for other collision avoidance algorithms of static and dynamic obstacles after proper modifications. Optimisation-based verification process is selected to verify the obstacle avoidance systems. In addition, decision-making collision avoidance algorithm is an attractive method for moving UAVs. This algorithm will be developed and verified within the parameters uncertainties.

Chapter 3

Uncertainty Analysis of Obstacle Avoidance Systems

3.1. Introduction

In the development of collision avoidance algorithms, only a simple kinematic model of the vehicle is used normally. This greatly simplifies the analysis and design of collision avoidance algorithms. However, the model in the verification stage must be as close to the real world as possible, which demands a much more complicated model. A simplified model of a vehicle and its operational environment is used in the algorithm development process while the real vehicle and its operational environment are much more complicated, with possibly a much high order of dynamics, nonlinearity, and much more complicated operation scenarios. This causes *structural uncertainties* in the verification of collision avoidance algorithms. The *parameter uncertainties* represent the variations of parameters that capture the changes of vehicle dynamics and its operational environment. The variations of the autonomous vehicle dynamics in operation may arise due to the changes in the vehicle itself (e.g. the change of mass or the centre of gravity) or the change of the operation environment (e.g. tyre friction for different road surfaces). In the online motion planning, unmanned vehicles must be able to sense obstacles, determine the obstacles positions and velocities, and reach the target position. However, there is inevitably *uncertainty in the sensor data* due to the limited accuracy of the robot's sensors and environmental noises. Therefore, it is necessary to verify whether or not an OAS under question is able to avoid obstacles with uncertain sensor data.

This chapter proposes an obstacle avoidance algorithm and control framework for UGV and UAV. In Chapter 1, a novel optimisation-based verification process of OAS for unmanned vehicles is defined to find the worst-case parameters and worst-case condition. This chapter presents the nominal and robustness analysis of OAS. The first step of verification process is to analysis the vehicle model and obstacle avoidance algorithm. After familiar with vehicle model and obstacle avoidance algorithm, the performance of proposed controller and obstacle avoidance algorithms are checked at nominal parameters. If it is satisfied the clearance criterion, then the next step is to study the robustness analysis for the vehicle OAS. Optimisation-based worst-case analysis is considered in the next chapters.

The artificial potential field method is chosen as a path planning and obstacle avoidance candidate technique for verification study for static and moving obstacles in 2D and 3D environments. Four case studies are presented in this chapter. In order to understand the verification process, first very simple unicycle robot is considered, and only two uncertain parameters are selected within the lower and upper bounds. Then this verification work is extended to the more complicated Pioneer 3-DX unicycle robot in the dynamic environment, and eight uncertain parameters, including sensor uncertainties are considered. After that, this work is extended to UAV static and moving OAS. Based on a 6 Degree of Freedom (6DoF) kinematic and dynamic model of a UAV, the path planning and collision avoidance algorithms are developed in 3D space, and three uncertain parameters are considered.

3.2 Case Study-1

3.2.1 Unicycle Mobile Robot Model

A schematic figure of a unicycle mobile robot is shown in Fig.3.1. This type of robot is mostly used for indoor applications. The robot moves in a global (X, Y) Cartesian co-ordinate plane and is represented by the following kinematic model with associated nonholonomic constraints (that disallows the robot from sliding sideways) [88, 89].

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\end{bmatrix} = \begin{bmatrix} cos\theta & 0\\ sin\theta & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix} v\\ \omega\end{bmatrix} \tag{3.1}$$

where v and ω are linear and angular velocities of the robot. Note that $[v, \omega]^T$ defines the inputs of the kinematic system. (x , y) are the robot position coordinates, and θ represents the orientation of the robot.

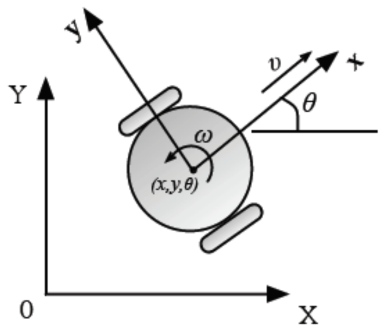


Figure 3.1: Unicycle Mobile Robot

The vehicle has two identical parallel, non-deformable rear wheels that are controlled by two independent motors and a steering front wheel. The dynamic equations of the unicycle mobile robot can be written as

$$\dot{v} = \frac{F}{m}$$
$$\dot{\omega} = \frac{\tau}{J} \tag{3.2}$$

where m is the robot's mass and J is the inertia moment. F and τ are the forward force input and moment torque input applied by the wheel motors respectively.

3.2.2 Motion Control

The controllers are proposed to have an inner-outer-loop structure (see Fig.3.2). The inner-loop control law is responsible to compute the force and torque signals that will tackle the wheel's motors to force the robot to move according to a desired linear and angular velocities. These desired velocities are the control signals generated by the outer-loop controller [90].

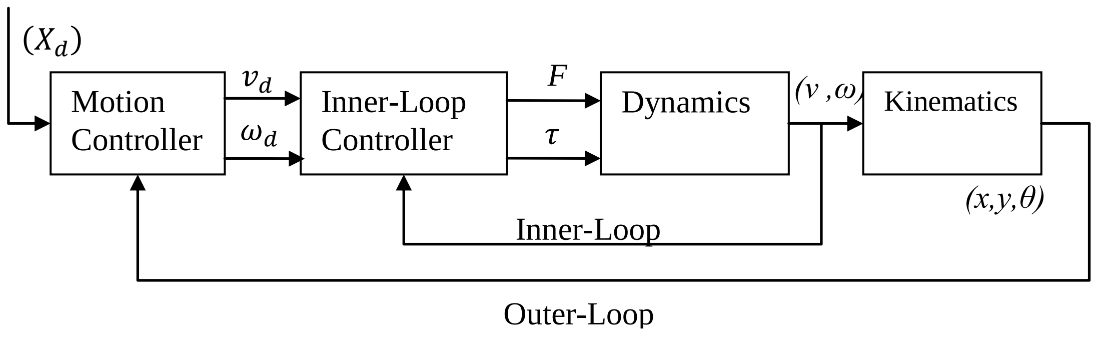


Figure 3.2: Model of the mobile-robot including kinematics, dynamics and the controllers

Inner-loop controller – PID Controller

To accomplish the goal of driving the robot to a desired linear velocity v_d and angular velocity ω_d , a first step is to compute the error between the true velocities and the desired ones. To this effect, let $e_v = v_d - v$ and $e_\omega = \omega_d - \omega$ be respectively the linear and angular velocity errors. A simple Proportional-Integral-Derivative (PID) control law is proposed as speed controllers.

$$F = K_{p1}e_v + K_{i1}\int_o^t e_v\,(\tau)d\tau + K_{d1}\frac{de_v}{dt}$$

$$\tau = K_{p2}e_\omega + K_{i2}\int_o^t e_\omega\,(\tau)d\tau + K_{d2}\frac{de_\omega}{dt} \tag{3.3}$$

Outer-loop controller - Motion planning for nonholonomic robots

The incremental motion planning for nonholonomic robot is considered in this section. In general, a kinematic model is used for motion planning and collision avoidance. The kinematic model of the wheeled mobile robot Eq. (3.1) can be represented in a general state space form as

$$\dot{X} = G(X)u \tag{3.4}$$

where $X \in \mathbb{R}^n$ is the vector of generalized coordinates, and $u \in \mathbb{R}^m$ $(m < n)$ is the control input vector.

Given any goal position and obstacle positions trajectory $X_d(t)$, a straightforward approach is to determine the input command u using the pseudo-inverse control law [91]

$$u = G^{\#}(X)\dot{X}_d \tag{3.5}$$

where $G^{\#}(X) = [G^T(X)G(X)]^{-1}G^T(X)$ is the pseudo-inverse of $G(X)$.

If the desired velocity $\dot{X}_d$ is feasible at the current X, the Eq. (3.5) will result in zero velocity error. A weighted pseudoinverse can also be used to balance error components. In Eq. (3.5), $\dot{X}_d$ can be chosen as the output of an incremental holonomic planner, and artificial potentials are used to drive the robot. The strategy is to modify the output of an incremental holonomic planner, and generates velocity control inputs that realize the desired motion in a least-squares sense [91, 92].

For the unicycle robot, $X = [x, y, \theta]^T$ is the configuration vector. Comparing Eqs. (3.1) and (3.4),

$$G(X) = \begin{bmatrix} cos\theta & 0 \\ sin\theta & 0 \\ 0 & 1 \end{bmatrix} \tag{3.6}$$

Let $u=[u_1, u_2]^T$, where u_1 is the linear velocity and u_2 is the angular velocity.

It follows from Eq. (3.5) that the pseudo-inverse of $G(x)$ takes the form

$$G^{\#}(X) = \begin{bmatrix} cos\theta & sin\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.7}$$

and the feedback law Eq. (3.5) for tracking a desired trajectory $X_d = [x_d , y_d , \theta_d]^T$ becomes

$$u = G^{\#}(X)\dot{X}_d = \begin{bmatrix} cos\theta & sin\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{x}_d \\ \dot{y}_d \\ \dot{\theta}_d \end{bmatrix} \tag{3.8}$$

Therefore, the resulting input command will be

$$u_1 = v_d = k_p(\dot{x}_d cos\theta + \dot{y}_d sin\theta)$$

$$u_2 = \omega_d = k_\theta \dot{\theta}_d \tag{3.9}$$

where gains k_p and k_θ are introduced for additional freedom in weighting the two input commands. This is equivalent to use a weighted pseudoinverse in Eq. (3.5). In order to apply the control law Eq. (3.9), the desired values $\dot{x}_d$, $\dot{y}_d$, and $\dot{\theta}_d$ are required. $\dot{X}_d$ may be determined by using the potential field method as described in the next section.

3.2.3 Static Obstacle Avoidance using with Potential Field Method

The artificial potential field method is one of the most common techniques in obstacle avoidance for mobile robots and manipulators. In this technique, a robot acts as a positive charge which is attracted by negatively charged goal position, while obstacles act as positive charges generating repulsive forces that push the robot away from the obstacles [64]. The combination of the attractive force to the goal and repulsive forces away from the obstacles drive the robot in a safe path to the goal.

The classical attractive potential function is

$$U_{att}(q) = \frac{1}{2} k_{att} d_{goal}^2(q) \tag{3.10}$$

where $q=[x, y]^T$, $d_{goal}= ||q - X_{goal}||$ is the Euclidean distance of the robot position q to the goal position X_{goal}. k_{att} is a scaling factor. In Fig 3.3, an attractive potential field is created using Eq. (3.10). The goal is positioned at (6, 7). k_a is chosen as 5.

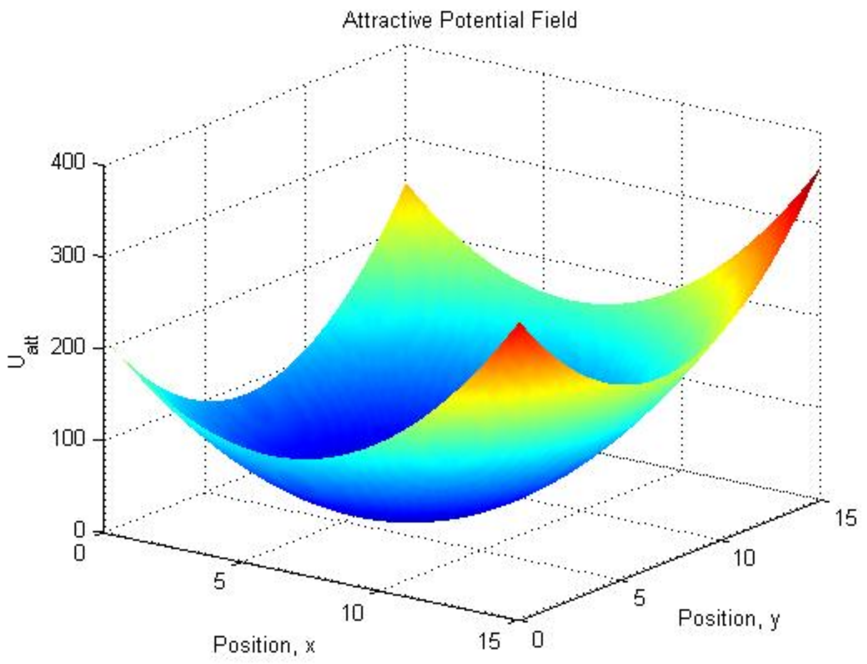


Figure 3.3: Attractive Potential Field

The attractive force is the negative gradient of the attractive potential

$$F_{att}(q) = -\nabla U_{att}(q) = -k_{att}(q - X_{goal}) \tag{3.11}$$

The repulsive potential function is

$$U_{rep_i}(q) = \begin{cases} \frac{1}{2} k_{rep} \left(\frac{1}{d_{obst_i}(q)} - \frac{1}{d_0} \right)^2, & \text{if } d_{obst_i}(q) \le d_0 \\ 0 & , \text{ if } d_{obst_i}(q) > d_0 \end{cases} \tag{3.12}$$

where $d_{obst_i}(q)$ is the closest distance to the obstacle i, k_{rep} is a scaling constant and d_0 is the influence threshold of the obstacle [64; 70]. In Fig 3.4, the repulsive potential field is created using Eq. (3.12). The obstacle is located at (6, 7).

The negative gradient of the repulsive potential $F_{rep_i}(q) = -\nabla U_{rep_i}(q)$ is given by,

$$F_{rep_i}(q) = \begin{cases} k_{rep}\left(\frac{1}{d_{obst_i}(q)} - \frac{1}{d_o}\right)\left(\frac{1}{d^2_{obst_i}(q)}\right)\frac{\partial d_{obst_i}(q)}{\partial (q)} & , \text{ if } \quad d_{obst_i}(q) \leq d_o \\ 0 & , \text{ if } \quad d_{obst_i}(q) > d_o \end{cases} \tag{3.13}$$

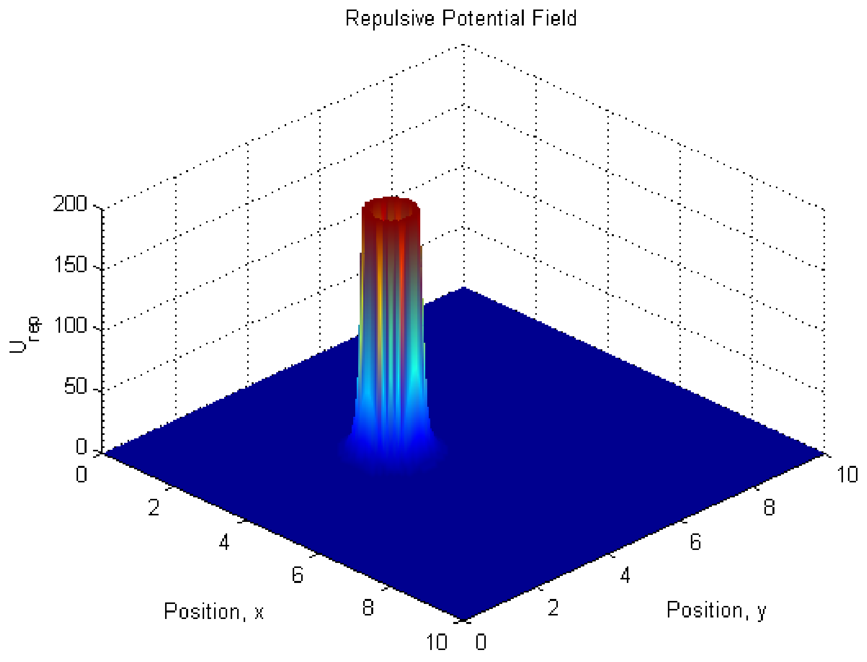


Figure 3.4: Repulsive Potential Field

Therefore, the total potential field (See Fig.3.5) is obtained by adding the repulsive potential resulting from all obstacles and the attractive potential from the target as given below:

$$F_{Total}(q) = F_{att}(q) + F_{rep_i}(q) \tag{3.14}$$

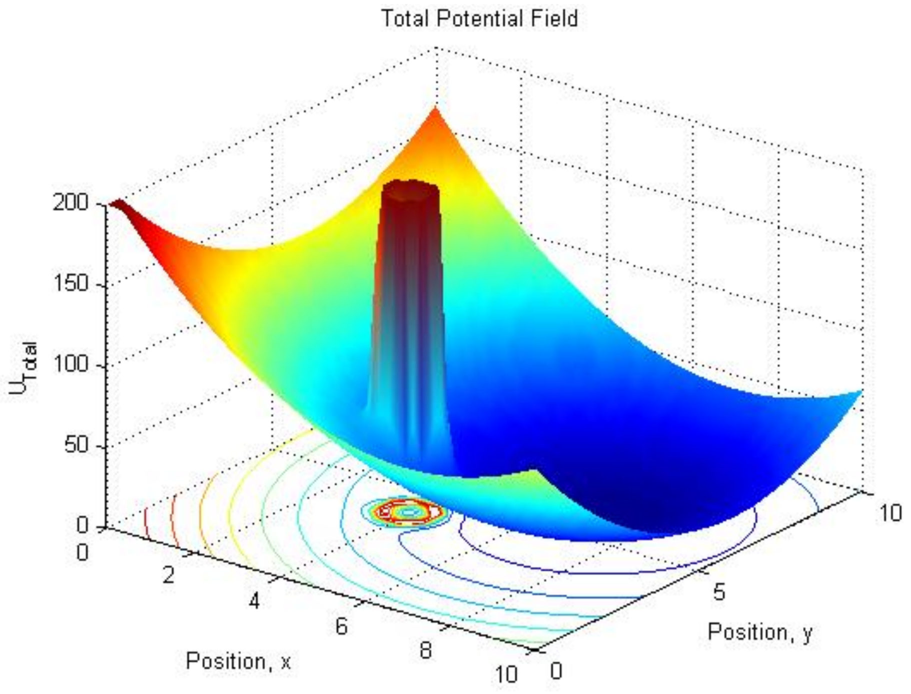


Figure 3.5: Total Potential Field

The desired velocities $\dot{x}_d$, $\dot{y}_d$ and $\dot{\theta}_d$ are selected as the natural motion in quasi-static conditions arising from the above force field. Therefore,

$$\begin{bmatrix} \dot{x}_d \\ \dot{y}_d \end{bmatrix} = F_{att}(q) + F_{rep_i}(q) \tag{3.15}$$

The rotational part of $\dot{\theta}_d$ is defined as

$$\dot{\theta}_d = atan2\left\{\frac{\dot{y}_d}{\dot{x}_d}\right\} - \theta \tag{3.16}$$

By defining *atan2{0,0}=θ*, the above function remains continuous along any approaching direction to the goal. The resulting command v_d and ω_d are determined by Eqs. (3.9), (3.15) and (3.16).

3.2.4 Analysis at nominal case

In this section, the simulation results at nominal parameters for a unicycle mobile robot among circular obstacles in a two-dimensional workspace are presented. The nominal parameter values are $m=5kg$ and $J=0.05kgm^2$. Controller gains are set to $k_p=0.06$, $k_\theta=5$, $Kp_1=6$, $Kp_2=5$, $K_{i1}=0.05$, $K_{i2}=0.05$, $K_{d1}=0.5$, $K_{d2}=0.5$, while the holonomic planner parameters are $k_a=5$, $k_r=4$. The influence range d_0 is chosen as 2*m*. The target position is located at (6, 7), and the obstacle is located at (4, 4) with a safety radius (r) of $0.5m$. The robot starts from initial position (0, 0).

The simulation result is shown in Fig. 3.6. The minimum distance to the obstacle is obtained as 0.9436*m* which is greater than the safety radius 0.5*m* $(d_{\min} > r)$. Therefore, the controller and obstacle avoidance algorithm is working correctly at nominal parameters.

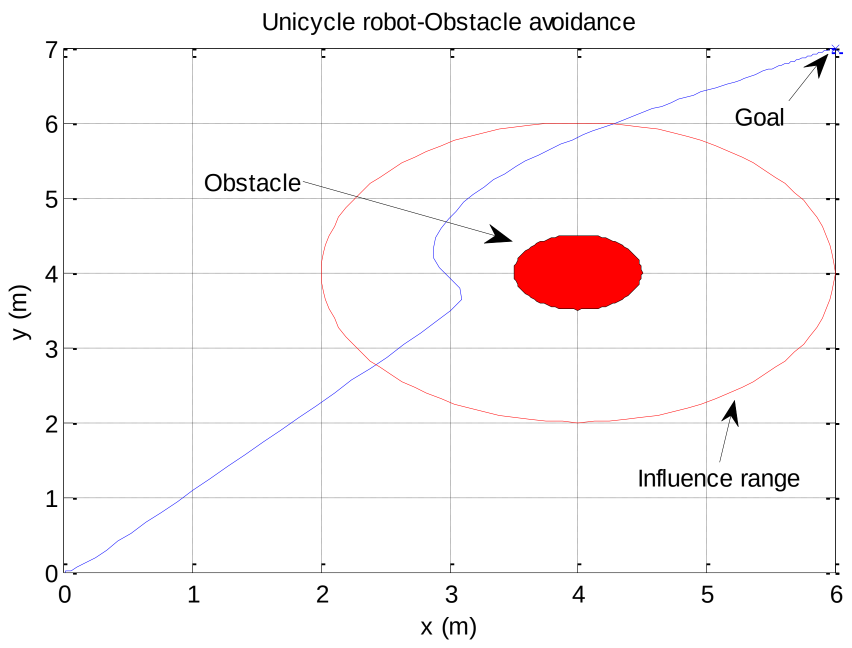


Figure 3.6: Simulation result for unicycle robot with obstacle at nominal parameters

3.2.5 Study on the effect of uncertainties

In Chapter 1, the anti-collision avoidance condition is presented for the circular obstacle i.e. if the minimum distance to the obstacle is greater than the safety radius of obstacle *($d_{min} > r$),* then no collision (Recall the Eq. (1.1)).

Initial robustness analysis of the proposed algorithm is carried out. Uncertainties are considered in the dynamic model, and each uncertain parameter is allowed to vary within *±20%* of its nominal value. Two uncertain parameters mass (*m*) and inertia (*J*) are considered within lower and upper bounds, i.e. *m= [4, 6] kg*, and *J= [0.04, 0.06] kgm*^{2}. Figs. 3.7 - 3.8 show variations of the minimum distance to the obstacle with respect to the mass and inertia. There is a small variation in the distance with the variations of the mass, but in a nonlinear form, whereas the minimum distance to the obstacle monotonically decreases with the increase of the inertia. In this case study, the simplified dynamic model equations of the mobile robot which is defined in Eq. (3.2) is used. This causes structural uncertainties in the verification of collision avoidance algorithms. Therefore, the mass causes small variation in a nonlinear form. In case study-2, the much more complicated unicycle robot model is used as close to the real world as possible.

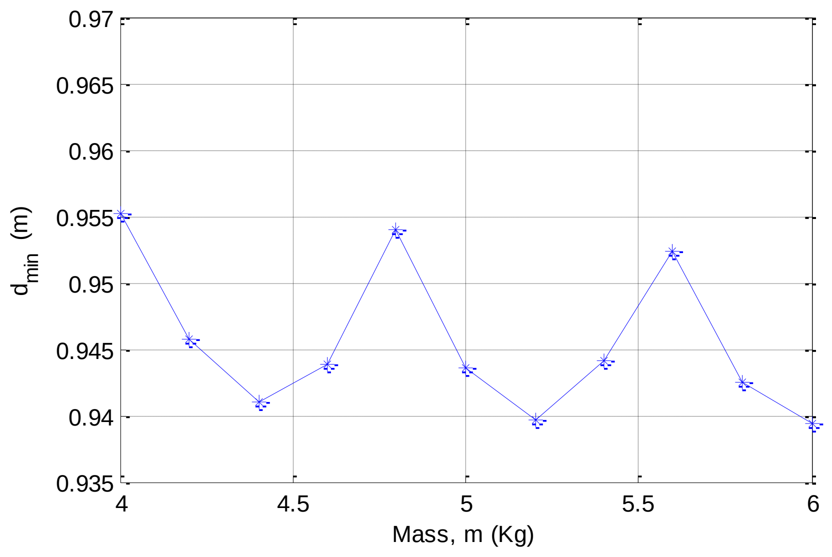


Figure 3.7: Mass variations in 20% range

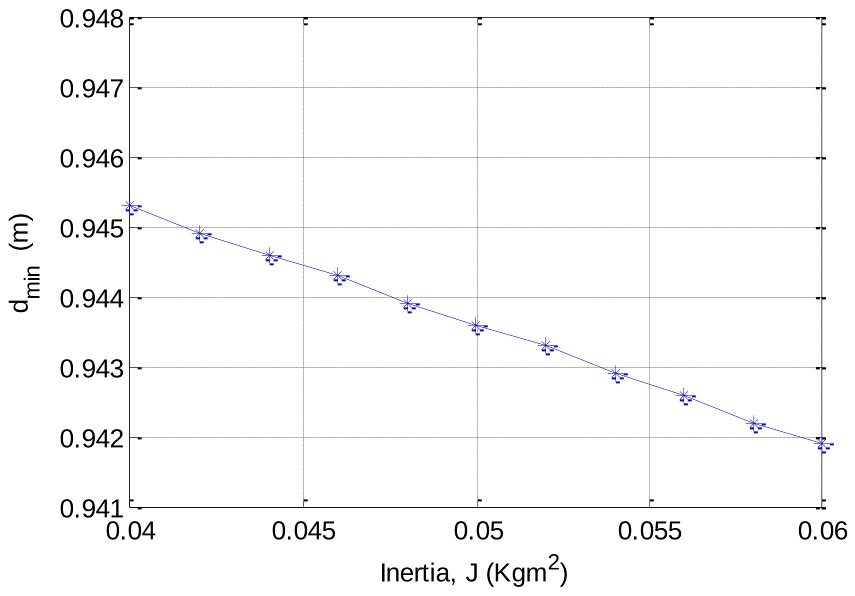


Figure 3.8: Inertia J- variations in 20% range

3.3 Case Study-2

3.3.1 Pioneer 3-DX Mobile robot model

Pioneer 3-DX (See Fig.3.9) is an intelligent mobile robot. It can carry loads more robustly. P3-DX has been used in many applications including automating highway maintenance and constructions. The robot mass is 9kg with the payload of 25kg. A schematic figure of a unicycle-like mobile robot is shown in Fig.3.10 [106].

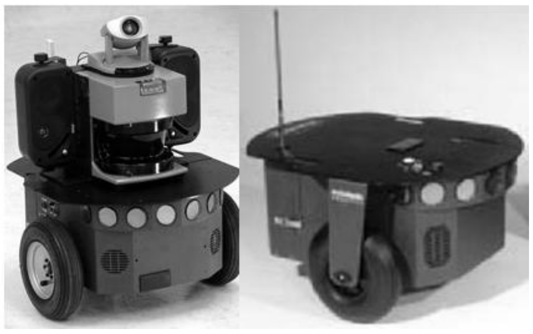


Figure 3.9: Pioneer-3DX mobile robots [44]

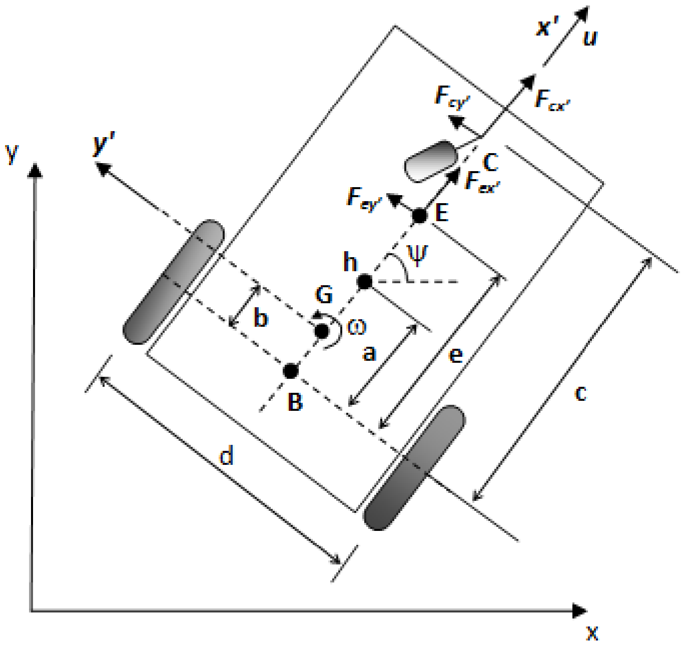


Figure 3.10: Parameters of the unicycle-like mobile robot

where *G* is the centre of mass; $h=[x\ y]^T$ is the point that is required to track a trajectory; *u* is the longitudinal and lateral velocities of the centre of mass; ω and ψ are the angular velocity and heading of the robot respectively; *D, b, a, e* and *c* are various distances ad defined in the figure; *C* is the position of the caster wheel; $F_{cx'}$ and $F_{cy'}$ are the longitudinal and lateral force exerted on *C* by the caster wheel respectively; *E* is the location of a tool onboard the robot; $F_{ex'}$ and $F_{ey'}$ are the longitudinal and lateral force exerted on *E* by the tool respectively [106].

In the robotic industry, most robots have low-level PID velocity controllers to track input reference velocities and the motor voltage $(V_u\ ,\ V_\omega)$ is not driven directly. Therefore, linear and angular reference velocities are considered as control signals [106]. In order to express these control signals, the robot servos have PD controllers to control the velocities of each motor. The corresponding proportional gains k_{PT} and k_{PR} , and derivative gains k_{DT} and k_{DR} are described in Eq. (3.17). These PD controllers are included in the model structure, which is shown in Fig.3.13.

$$\begin{bmatrix} V_u \\ V_\omega \end{bmatrix} = \begin{bmatrix} k_{PT}(u_{ref} - u) - k_{DT}\dot{u} \\ k_{PR}(\omega_{ref} - \omega) - k_{DR}\dot{\omega} \end{bmatrix} \tag{3.17}$$

The complete mathematical model is written as

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \\ \dot{u} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} u cos\psi - a\omega sin\psi \\ u sin\psi + a\omega cos\psi \\ \omega \\ \frac{\theta_3}{\theta_1}\omega^2 - \frac{\theta_4}{\theta_1}u \\ -\frac{\theta_5}{\theta_2}u\omega - \frac{\theta_6}{\theta_2}\omega \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{1}{\theta_1} & 0 \\ 0 & \frac{1}{\theta_2} \end{bmatrix} \begin{bmatrix} u_{ref} \\ \omega_{ref} \end{bmatrix} + \begin{bmatrix} \delta_x \\ \delta_y \\ 0 \\ \delta_u \\ \delta_\omega \end{bmatrix} \tag{3.18}$$

where u and ω are current robot linear and angular velocities; u_{ref} and ω_{ref} are linear and angular reference velocities; $\theta = [\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6]^T$ is the vector of model parameters, which are given below:

$$\theta_1 = \left[\frac{R_a}{k_a}(mR_t r + 2I_e) + 2rk_{DT}\right] / (2rk_{PT})$$

$$\theta_2 = \left[\frac{R_a}{k_a}(I_e d^2 + 2R_t r(I_z + mb^2)) + 2rdk_{DR}\right] / (2rdk_{PR})$$

$$\theta_3 = \frac{R_a}{k_a} mbR_t / (2k_{PT})$$

$$\theta_4 = \frac{R_a}{k_a}(\frac{k_a k_b}{R_a} + B_e) / (rk_{PT}) + 1$$

$$\theta_5 = \frac{R_a}{k_a} mbR_t / (dk_{PR})$$

$$\theta_6 = \frac{R_a}{k_a}(\frac{k_a k_b}{R_a} + B_e)d / (2rk_{PR}) + 1 \tag{3.19}$$

$\delta = [\delta_x, \delta_y, 0, \delta_u, \delta_\omega]^T$ is the uncertainty vector associated to the mobile robot:

$$\delta_x = -\bar{u}^s \sin\psi$$

$$\delta_y = \bar{u}^s \cos\psi$$

$$\delta_u = \frac{\frac{m\omega\bar{u}^s + F_{ex'} + F_{cx'}}{I_e} + \left(\frac{k_a k_b}{R_a} + B_e\right)\frac{u_r^s + u_l^s}{R_t r I_e} + \frac{\dot{u}_r^s + \dot{u}_l^s}{R_t r}}{\frac{m}{I_e} + \frac{2}{R_t r} + \frac{2k_{DT}k_a}{R_a R_t I_e}}$$

$$\delta_\omega = \frac{\left(\frac{k_a k_b}{R_a} + B_e\right)\frac{u_r^s - u_l^s}{2R_t r I_e} + \frac{\dot{u}_r^s - \dot{u}_l^s}{2R_t r} - \frac{mb\dot{\bar{u}}^s}{I_e d} + \frac{eF_{ey'} + cF_{cy'} + \tau_e}{I_e d}}{\frac{d}{2R_t r} + \frac{I_z + mb^2}{I_e d} + \frac{k_{DR}k_a}{R_t R_a I_e}} \tag{3.20}$$

where m is the robot mass; I_z is the robot moment of inertia about vertical axis located in G; r is the right and left wheels' radius; u_r^s and u_l^s are the longitudinal slip speeds of the right and left wheels; $\bar{u}^s$ is the lateral slip speed of the wheels; k_a is the torque constant multiplied by the gear ratio; k_b is the voltage constant multiplied by the gear ratio; k_t is the nominal radius of the tire; R_a is the electric resistance constant; I_e and B_e , respectively, are the moment of inertia and the viscous friction coefficient of the combined motor rotor, gearbox, and wheel; τ_e is the moment exerted on E by the tool [106].

3.3.2 Clearance criterion for Moving Obstacle Avoidance

The motion planning of a mobile robot in a dynamic environment is to plan and control the robot motion from the starting position to the goal position while avoiding moving obstacles. The dynamic obstacle avoidance algorithm in 2D is investigated in this study where a potential field-based dynamic obstacle avoidance algorithm for non-cooperative robot is selected.

As shown in Fig.3.11, ρ_0 is a positive constant describing the potential field influence range of the obstacle. In general, one robot is considered as an 'intruder' (Robot-B) whereas the other one (Robot-A) is assumed to be equipped with an OAS which is capable of detecting and avoiding the intruder without knowing its intention.

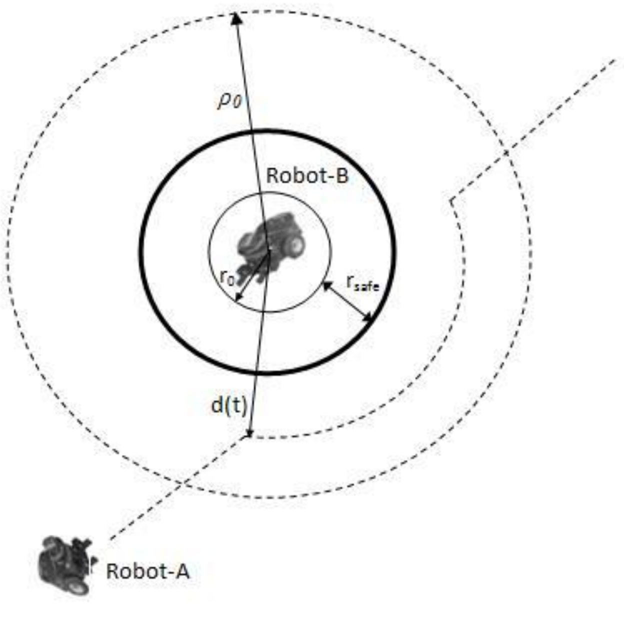


Figure 3.11: Moving obstacle avoidance clearance criterion

In Fig.3.12, the relative velocity between the robot and the obstacles in the direction from the robot to the obstacle is defined as [74]:

$$v_{RO}(t) = [v(t) - v_{obs}(t)]^T n_{RO} \tag{3.21}$$

where n_{RO} is a unit vector pointing from the robot to the obstacle; $v(t)$ and $v_{obs}(t)$ are the robot and obstacle velocities respectively. If $v_{RO}(t) \leq 0$, then the robot is moving away from the obstacle. Therefore, no avoidance manoeuvre is needed. If $v_{RO}(t) > 0$, the robot is moving close to the obstacle and avoidance manoeuvre must be activated when the distance between the vehicle and the obstacle is predicted to be below a certain threshold.

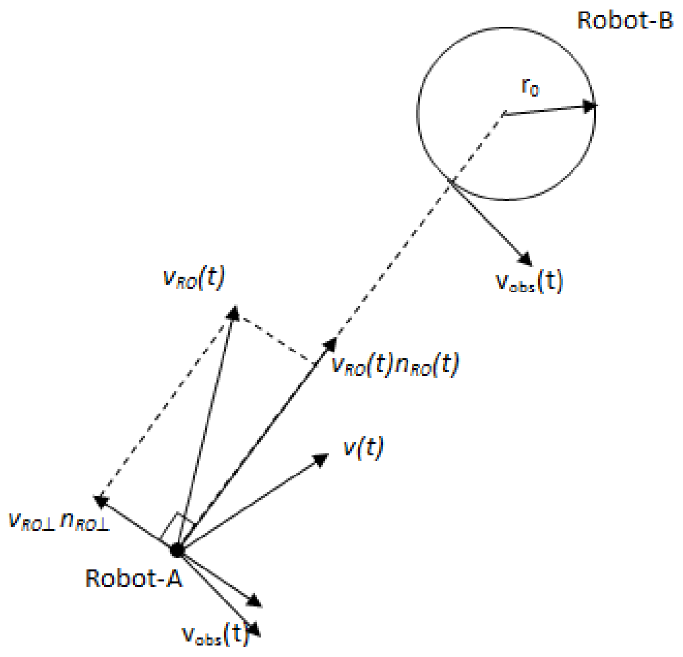


Figure 3.12: Relative velocity between the robot and the obstacle

The minimum distance to the obstacle (d_{min}) is defined as the clearance criterion in the time domain. Robot-A can detect the moving obstacle's shape, positions, orientation and velocity, where a moving obstacle is considered as a circular object. For a moving OAS safety analysis, an intruder is defined with a radius of r_0 and a safety margin of r_{safe}. The intruder radius and safety margin can be chosen according to the dimensions of robots.

Letting $r = r_0 + r_{safe}$, the anti-collision condition is defined as $d_{min}>r$. In a moving OAS process, all violations of the clearance criteria must be found and corresponding worst-case combination of uncertain parameters must also be computed.

3.3.3 Motion Control and Obstacle Avoidance

The control system involves two control loops (inner and outer) as shown in Fig.3.13. The outer-loop is the motion controller which generates the desired linear velocity u_d and angular velocity ω_d. The inner-loop is chosen as a Proportional-Integral (PI) controller because the robot servos already have built-in PD controllers to control the velocities of each motor. The inner-loop PI control law is responsible to compute the linear and angular reference velocities signals (u_{ref} and ω_{ref}). True and desired velocities are saturated without exceeding given limits.

Inner-Loop Controller

A PI control law with anti-windup is proposed as speed controllers which are given below. The goal of the inner loop is to achieve and maintain the desired linear velocity u_d and desired angular velocity ω_d.

$$\begin{aligned} u_{ref} &= K_1 e_u + K_3 \int_o^t e_u(\tau)d\tau \\ \omega_{ref} &= K_2 e_\omega + K_4 \int_o^t e_\omega(\tau)d\tau \end{aligned} \tag{3.22}$$

where $K_1$and K_2 are proportional controller gains, and K_3 and K_4 are integral controller gains. $e_u = u_d - u$ and $e_\omega = \omega_d - \omega$ are the linear and angular velocity errors respectively.

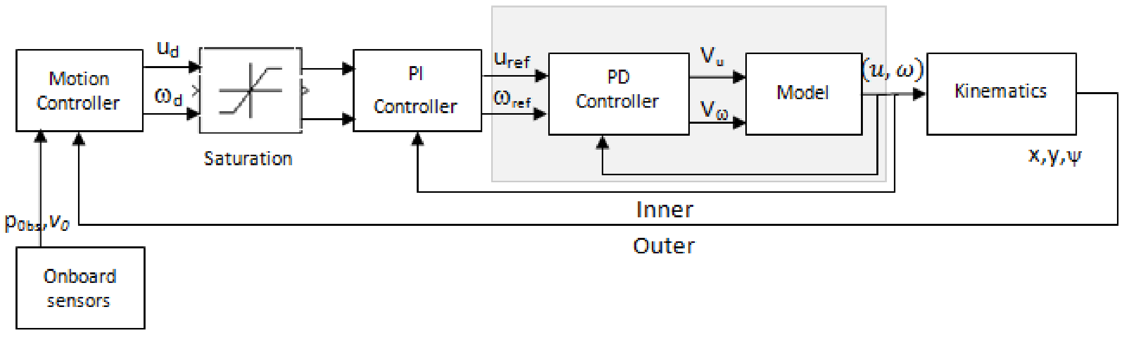


Figure 3.13 Mobile robot motion planning control systems

Outer-Loop Motion Controller

Recall the Eq. (3.4) and (3.5). For the Pioneer unicycle robot, $X = [x, y, \psi]^T$ is the configuration vector.

$$G(X) = \begin{bmatrix} cos\psi & -asin\psi \\ sin\psi & acos\psi \\ 0 & 1 \end{bmatrix} \tag{3.23}$$

Let $U=[u_d, \omega_d]^T$. The pseudo-inverse of *G(X)* takes the form

$$G^{\#}(X) = \frac{1}{(a^2+1)} \begin{bmatrix} (a^2+1)cos\psi & (a^2+1)sin\psi & 0 \\ -asin\psi & acos\psi & 1 \end{bmatrix} \tag{3.24}$$

Correspondingly, the feedback law for tracking a desired trajectory $X_d = [x_d, y_d, \psi_d]^T$ becomes

$$U = \frac{1}{(a^2+1)} \begin{bmatrix} (a^2+1)cos\psi & (a^2+1)sin\psi & 0 \\ -asin\psi & acos\psi & 1 \end{bmatrix} \begin{bmatrix} \dot{x}_d \\ \dot{y}_d \\ \dot{\psi}_d \end{bmatrix} \tag{3.25}$$

which can be written as

$$u_d = k_p(\dot{x}_d cos\psi + \dot{y}_d sin\psi) \tag{3.26}$$

$$\omega_d = \frac{k_q}{(a^2+1)}(-a\dot{x}_d sin\psi + a\dot{y}_d cos\psi + \dot{\psi}_d) \tag{3.27}$$

where gains *kp* and k_q are introduced to allow for additional freedom in weighting the input commands. In order to apply the control law Eq. (3.26) and (3.27), the desired velocities have to be specified. These desired values can be determined using the potential field method as described in the next section.

3.3.4 Moving Obstacle Avoidance using with Potential Field Method

Previous studies use potential field methods to deal with robot path planning in static environments where obstacles are all stationary. However, the environments in real-time applications are dynamic. In [74], the potential field method for motion planning of a mobile robot in a dynamic environment was proposed. The attractive potential field is defined as a function of the robot position to the goal position. The repulsive potential is defined as the function of the relative position and velocity of the robot with respect to the moving obstacles. The virtual force is defined as the negative gradient of the potential field. In this approach, the motion planning only needs the online sensor measurements of the moving obstacles' information. To simplify the analysis, the following assumptions are made:

Assumptions 1: The shape, position h, and velocity v of the robot are known.
Assumptions 2: The target position p_{tar} is known.
Assumptions 3: the obstacles' shapes, positions p_{obs} , and velocities v_{obsi} can be measured online.

Attractive Potential Function: The attractive potential field is defined as a function of the robot position to the target position where the target is a fixed point in space. The attractive potential field function is defined as follows:

$$U_{att}(h) = \alpha_p \|p_{tar} - h\|^2 \tag{3.28}$$

The corresponding attractive force is defined as:

$$F_{att}(h) = \; k_{att}(p_{tar} - h\,) \tag{3.29}$$

where α_p and k_{att} are the positive constants; p_{tar} is the goal position; h is the robot position.

Repulsive Potential Function: A repulsive potential function is defined as relative positions and velocities between the robot and the obstacles. The relative velocity between the robot and the obstacle in the direction from the robot to the obstacle is given in Eq. (3.21). No avoidance motion is needed when $v_{RO}(t) \leq 0$ because the robot is moving away from the obstacle, but if $v_{RO}(t) > 0$, then the robot is moving close to the obstacle. Therefore, the avoidance motion has to be considered.

If a maximum deceleration magnitude A_{max} is applied to the robot to reduce its velocity, the distance travelled by the robot before v_{RO} defined in Eq. (3.21) reduces to zero is

$$\rho_m(v_{RO}) = \frac{v_{RO}^2(t)}{2A_{max}} \tag{3.30}$$

The repulsive potential is defined as follows:

$$U_{rep}(h,v) = \begin{cases} 0 & , \quad \text{if } \rho_s(h,p_{obs}) - \rho_m(v_{RO}) \geq \rho_0 \text{ or } v_{RO} \leq 0 \\ \eta\left(\frac{1}{\rho_s(h,p_{obs})-\rho_m(v_{RO})} - \frac{1}{\rho_0}\right), & \quad \text{if } 0 < \rho_s(h,p_{obs}) - \rho_m(v_{RO}) < \rho_0 \text{ and } v_{RO} > 0 \\ \text{not defined} & , \quad \text{if } v_{RO} > 0 \text{ and } \rho_s(h,p_{obs}) < \rho_m(v_{RO}) \end{cases} \tag{3.31}$$

where ρ_0 is a positive constant describing the influence range of the obstacle; η is a positive constant; v_{RO} is a relative velocity between the robot and the obstacle in the direction from the robot to the obstacle; ρ_s is the shortest distance between the robot and the body of the obstacle.

The velocity component perpendicular to $v_{RO}(t)n_{RO}$ (See. Fig.3.14) is given in the following equation

$$v_{RO\perp}n_{RO\perp} = v(t) - v_{obs}(t) - v_{RO}(t)n_{RO} \tag{3.32}$$

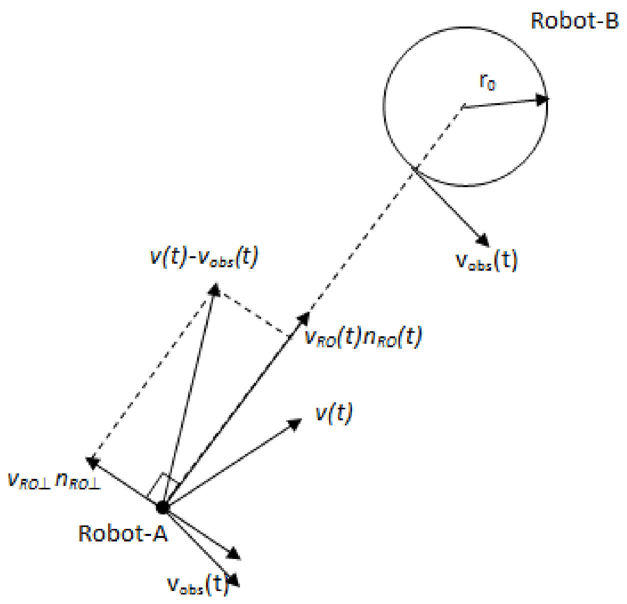


Figure 3.14: The velocity component perpendicular to $v_{RO}(t)n_{RO}(t)$

The corresponding repulsive force (See. Fig.3.15) is defined as the negative gradient of the repulsive potential in terms of both position and velocity

$$F_{rep}(h,v) = \begin{cases} 0 & , \quad \text{if } \rho_s(h,p_{obs}) - \rho_m(v_{RO}) \geq \rho_0 \ \text{ or } \ v_{RO} \leq 0 \\ F_{rep1} + F_{rep2} & , \quad \text{if } \ 0 < \rho_s(h,p_{obs}) - \rho_m(v_{RO}) < \rho_0 \ \text{ and } \ v_{RO} > 0 \\ \text{not defined} & , \quad \text{if } v_{RO} > 0 \ \text{ and } \ \rho_s(h,p_{obs}) < \rho_m(v_{RO}) \end{cases}$$

(3.33)

where

$$F_{rep1} = -\frac{\eta}{(\rho_s(h,p_{obs}) - \rho_m(v_{RO}))^2}\left(1 + \frac{v_{RO}}{A_{max}}\right)n_{RO} \tag{3.34}$$

and

$$F_{rep2} = \frac{\eta v_{RO} v_{RO\perp}}{\rho_s(h,p_{obs}) A_{max} (\rho_s(h,p_{obs}) - \rho_m(v_{RO}))^2} n_{RO\perp} \tag{3.35}$$

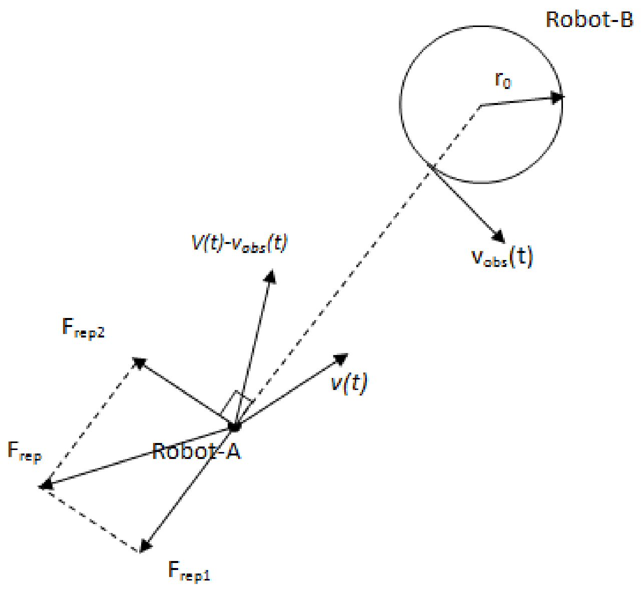


Figure 3.15: Repulsive forces

The total force F_{Total} is the combination of attractive force and repulsive force. The total virtual force is used for motion planning. More details can be found in [74].

Therefore,

$$\begin{bmatrix} \dot{x}_d \\ \dot{y}_d \end{bmatrix} = F_{Total} = F_{att} + F_{rep}(h, v) \tag{3.36}$$

$$\dot{\psi}_d = atan2\left\{\frac{\dot{y}_d}{\dot{x}_d}\right\} - \psi \tag{3.37}$$

By defining $atan2\{0,0\}=\psi$, the above function remains continuous along any approaching direction to the goal. The resulting command u_d and ω_d are determined by Eq. (3.26), (3.27), (3.36) and (3.37).

3.3.5 Simulation Results at Nominal Parameters

Simulation is carried out to confirm that a desirable performance is achieved at the nominal case under the design described in the previous sections. The nominal parameter values of the robot are given in Table.3.1 [71]. The uncertainty vector δ_{un} is considered as $[-0.05\sin\psi\ ,\ 0.05\cos\psi\ ,\ 0\ ,\ 0.2\ ,\ 0.5\ ]^T$. The PI controller gains and motion planner parameters for potential field force are also tuned and set to fixed values for the verification process. Proportional gains k_{PT} and k_{PR} are set to *11* and, derivative gains k_{DT} and k_{DR} are set to *0.1*. The saturation limits of true and desired values of linear and angular speeds of the mobile robot used in the simulations are *[0 , 1.6] (m/s)* and *[-3.5 , 3.5](rad/s)* respectively.

The safety radius including safe margin is chosen as *5m*. The simulation results at 10, 15, 20, and 40 sec are shown in Figs.3.16 - 3.19. The intruder moves to the goal position without any avoiding manoeuvres, while robot avoids the intruder and reaches to the goal position. The minimum distance to the obstacle is obtained as 7.668m, which is greater than the safety radius (*dmin* > *r*). Therefore, the moving obstacle avoidance algorithm functions correctly at nominal parameters.

TABLE 3.1
UNICYCLE MODE, NOMINAL PARAMETERS

Symbol	Parameters	Initial Values
m	Robot mass and payload	18 (kg)
I_Z	Robot moment of inertia	20 kg.m^2
R_t	Radius of the tire	0.14 (m)
r	Right and left wheel radius	0.0977 (m)
k_a	Torque constant multiplied by the gear ratio	0.8808 (N.m/A)
R_a	Electric resistance constant	0.71 (Ω)
I_e	Moment of inertia of the combined motor rotor, gearbox, and wheel	2 kg.m^2
B_e	Viscous friction of the combined motor rotor, gearbox, and wheel	0.8
k_b	Voltage constant multiplied by the gear ratio	0.8808 (V.s/rad)
d	Width of the robot	0.395 (m)
a	Distance to the point h	0.25 (m)
b	Position of center of mass	0.1 (m)
L	Length of the robot	0.445 (m)

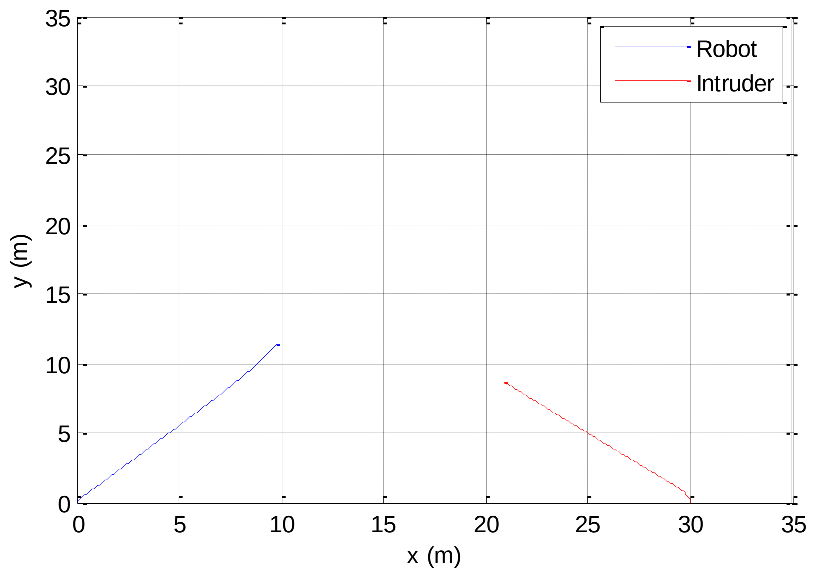


Figure 3.16: Simulation response at t=10 sec

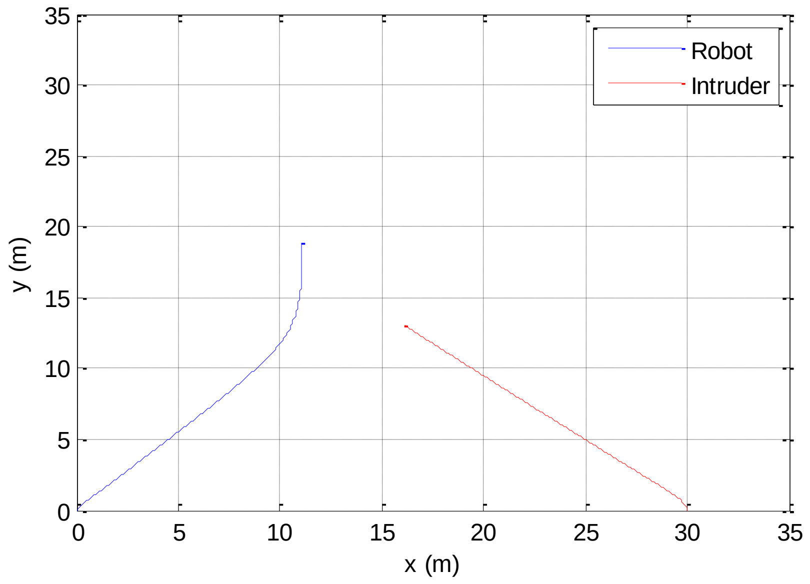


Figure 3.17: Simulation response at t=15 sec

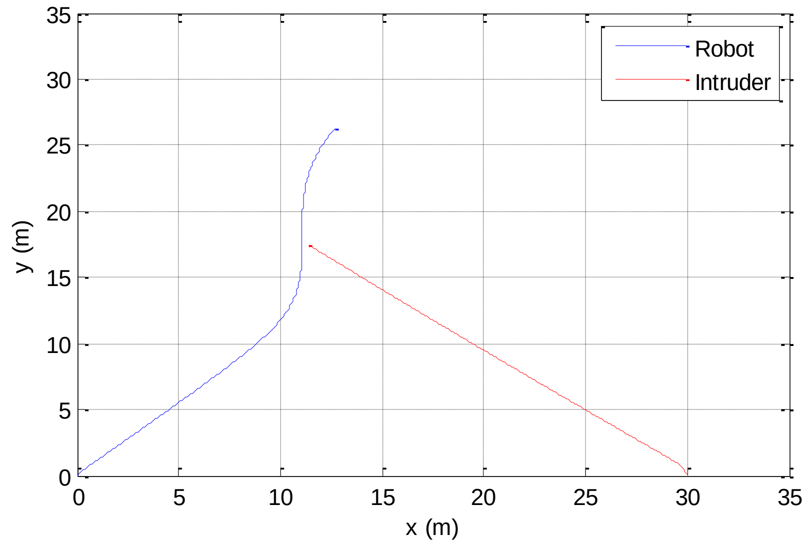


Figure 3.18: Simulation response at t=20 sec

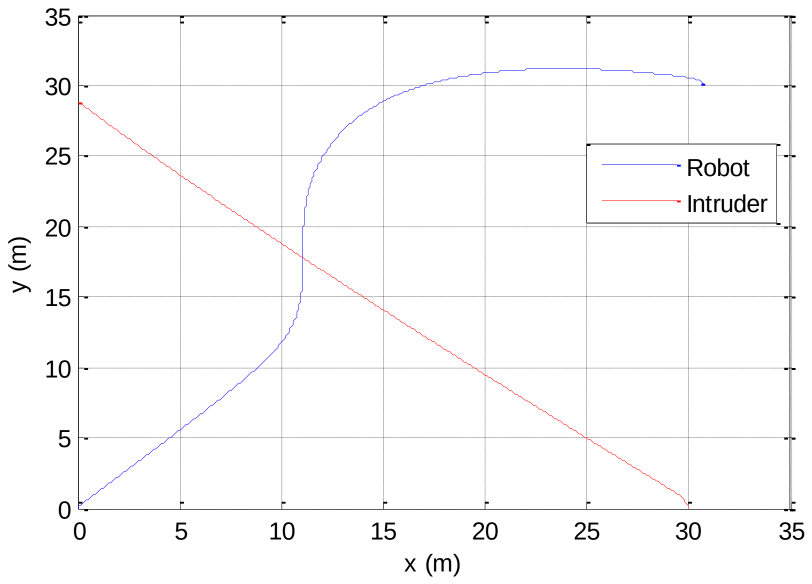


Figure 3.19: Simulation response at t=40 sec

3.3.6 Initial Robustness Analysis

Because kinematic and dynamic models involve many parameters, we questioned whether it is necessary to take into account all parameters in the worst-case analysis. There is no unique way to choose the parameters which may be important in a problem. The choice of the important parameters in the verification process depends on the problem. Initial robustness analysis can determine the most significant parameters that are most influence on the minimum distance to the obstacle (d_{min}). Uncertainties are introduced in the parameters, and each uncertain parameter is allowed to vary within ±10 or 20 % of its nominal value, i.e., uncertain parameters are considered within the lower and upper bounds. After that, within the uncertain parameter range, the minimum distance from the robot to an obstacle d_{min} during a collision avoidance manoeuvre is obtained from simulations. The uncertain parameter versus d_{min} responses are analysed and chosen the most significant parameters which influence on the minimum distance to the obstacle.

Initial robustness analysis of the proposed algorithm is carried out in this Section. Uncertainties are introduced in sensor data as follows: x position of the obstacle: $Px_0 = x_0 + \Delta x$; y position of the obstacle: $Py_0 = y_0 + \Delta y$; Obstacle orientation: $P\psi_0 = \psi_0 + \Delta\psi$; Obstacle velocity: $V_{v0} = v_{obs} + \Delta v$, where $p_{obs} = [x_0 , y_0 , \psi_0]^T$ is the true obstacle reading at the nominal case. Δx, Δy, and $\Delta\psi$ are sensor data errors in x_0, y_0 , and ψ_0 respectively. In the similar fashion, v_{obs} is the true obstacle velocity, and Δv is the velocity error. After analyzing the influence of obstacle detection sensor data uncertainties, the most significant uncertainties are found to be x and y position (i .e. Δx and Δy), which are chosen within the bounds to find the worst-case condition.

Eight uncertain parameters are considered in this case study. Lower and upper bounds of each uncertain parameter are given in Table.3.2. The structural uncertainty of δ_x, δ_y, δ_u and $\delta\omega$ are considered in this study. Variation in lateral slip speed ($\bar{u}^s$) is applied within the range for the uncertainty of δ_x and δ_y. All possible dynamic model parameter variations are considered, and most significant are selected for the optimisation search process to find the worst case. The clearance criterion of minimum distance to the obstacle depends on the proposed control laws and collision avoidance algorithm in the presence of all possible parameters variations. The kinematic model of the vehicle is used in the development of collision avoidance algorithm.

However, in the verification process, a much more complicated model including kinematic and dynamic model, the speed controller and motion controller are considered to find the worst cases. Therefore, finding the worse case is much more complicated and challenging task in the verification process of OAS. Moreover, prediction of minimum distance to the obstacle with respect to parameter variation is also complicated. However, the initial robustness analysis can determine the performance. Figs.3.20 -3.27 show the variations of the minimum distance to the obstacle with respect to parameter variations. It clearly shows that for different uncertain parameters, the influence of the minimum distance to the obstacle could be quite different. The minimum distance almost linearly depends on the variations of each parameter. The minimum distance to the obstacle decreases with increase of I_e , m , δ_u , and Δx.

TABLE 3.2
UNICYCLE MODEL, UNCERTAIN PARAMETERS

Parameter	Description	Bounds
Δx	Variation in sensor data, x	[-0.5, 0.5]
Δy	Variation in sensor data, y	[-0.5, 0.5]
m	Variation in robot mass and payload(kg)	[9, 34]
B_e	Variation in viscous friction of the combined motor rotor, gearbox, and wheel	[0.48, 1.12]
δ_u	Variation in uncertainty in the linear acceleration (m/s^2)	[0.1, 0.9]
δ_ω	Variation in uncertainty in the angular acceleration (rad/s^2)	[0.1, 0.9]
I_e	Variation in moment of inertia of the combined motor rotor, gearbox, and wheel (kg.m^2)	[0.2, 3.8]
$\bar{u}^s$	Variation in lateral slip speed (m/s)	[0.02,0.08]

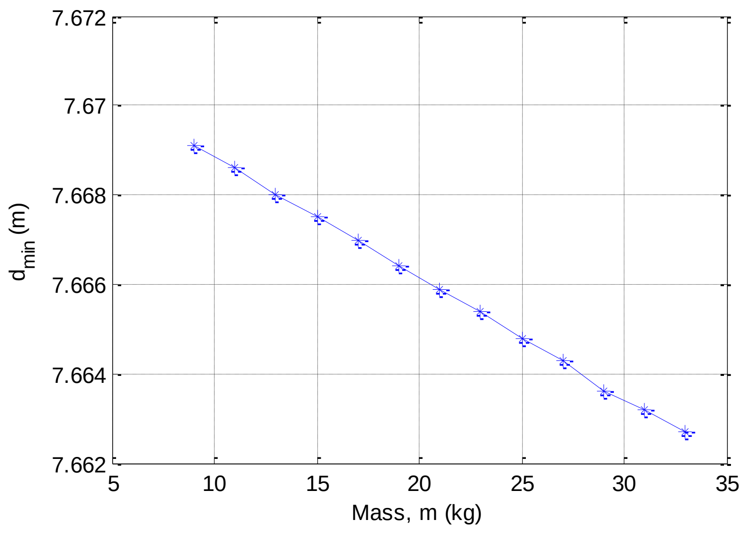


Figure 3.20: Variation in robot mass and payload, *m*

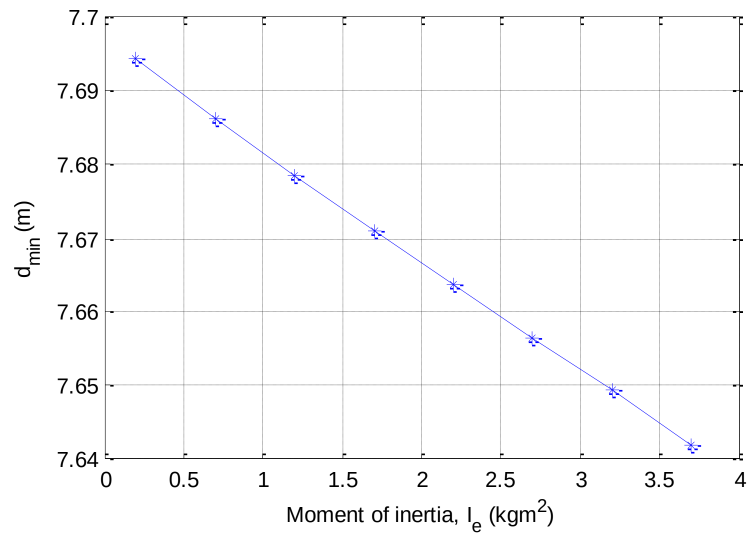


Figure 3.21: Variation in the moment of inertia, I_e

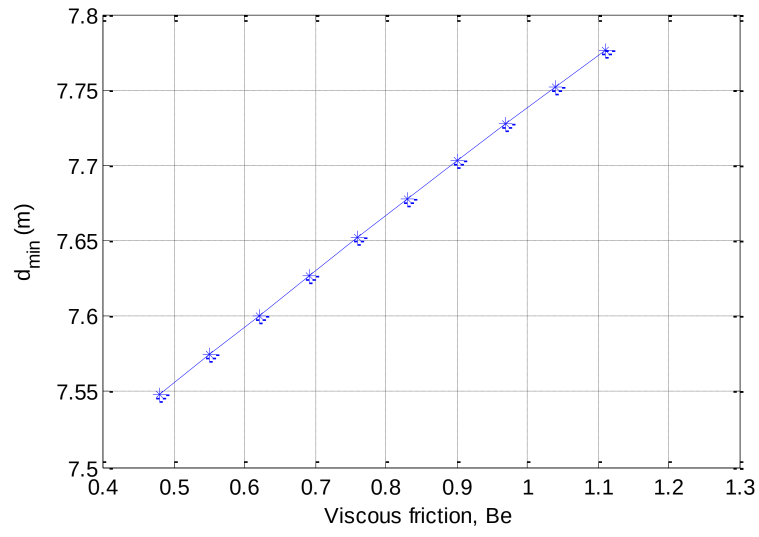


Figure 3.22: Variation in viscous friction, B_e

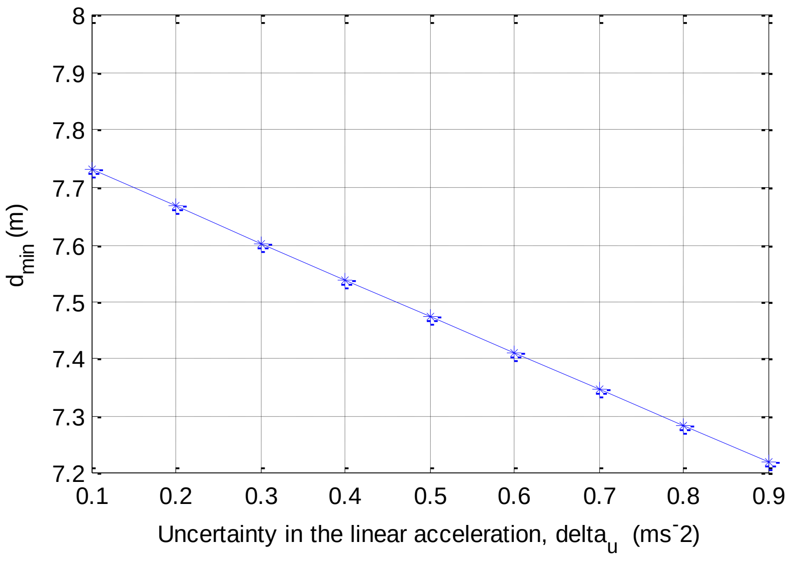


Figure 3.23: Variation in uncertainty in linear acceleration, δ_u

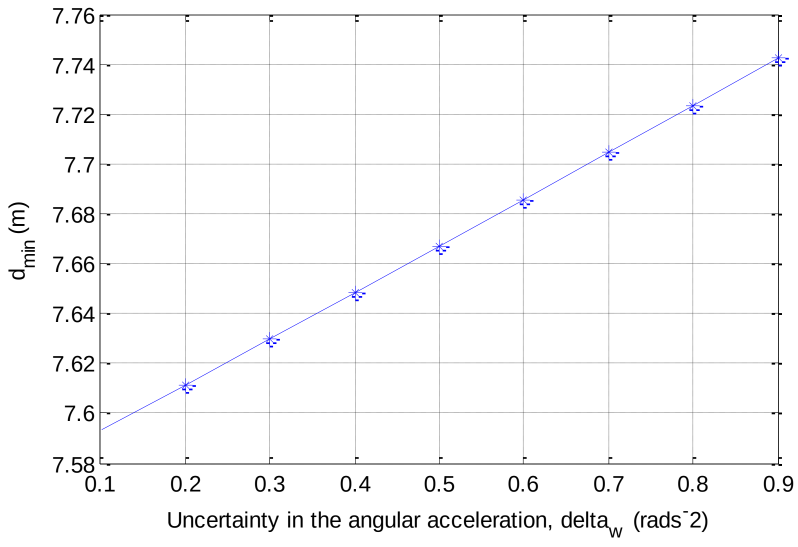


Figure 3.24: Variation in uncertainty in angular acceleration, $\delta\omega$

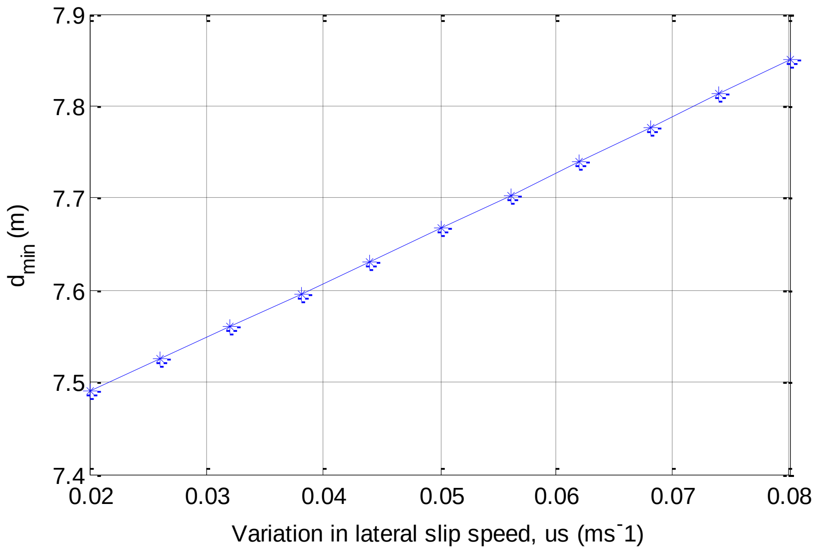


Figure 3.25: Variation in lateral slip speed of wheels, $\bar{u}^s$

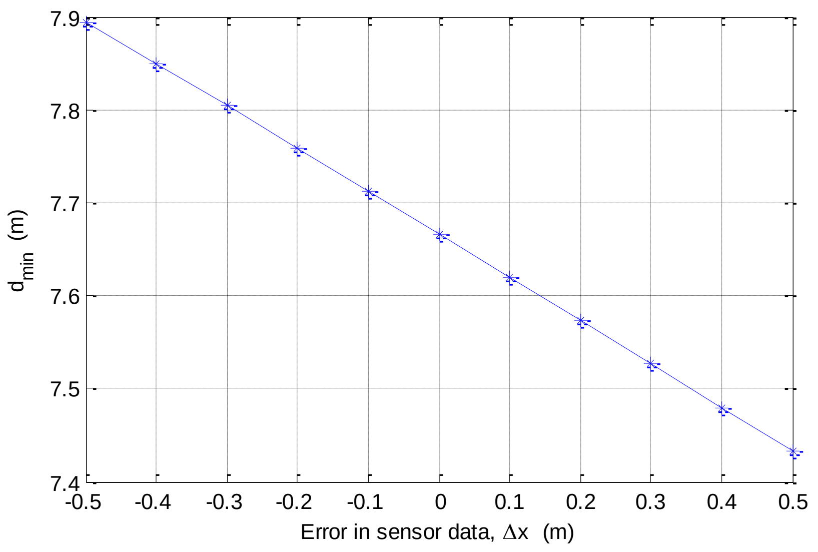


Figure 3.26: Variation in the sensor data, Δx

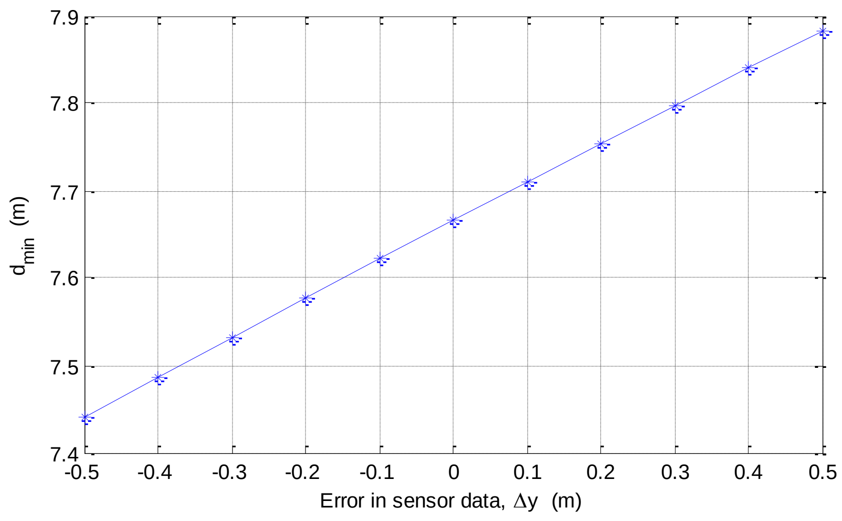


Figure 3.27: Variation in the sensor data, Δy

3.4 Case Study-3

3.4.1 UAV MODEL - Kinematic Equations

In order to present a clearance criterion of obstacle avoidance systems, an UAV model is considered for the algorithm development and assessment. Based on a 6 Degree of Freedom (6DoF) kinematic and dynamic model of a UAV, the path planning and collision avoidance algorithms are developed in 3D space. The configuration vector*s* = $[x, y, z, \phi, \theta, \psi]^T$ is used to specify the position and orientation of the UAV in the global coordination, where $q_o=[x,y,z]^T$ is the c.g. (center of gravity) position of the vehicle and $\boldsymbol{\varphi} = [\phi, \theta, \psi]^T$ are the Euler angles, with ϕ as the roll, θ as the pitch, and ψ as the yaw. Therefore, the absolute velocity in terms of the Euler angles and velocity components in the body frame can be written as follows [103-105]:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} c_\psi c_\theta & c_\psi s_\theta s_\phi - s_\psi c_\phi & s_\psi s_\phi + c_\psi c_\phi s_\theta \\ s_\psi c_\theta & c_\psi c_\phi + s_\phi s_\theta s_\psi & s_\theta s_\psi c_\phi - c_\psi s_\phi \\ -s_\theta & c_\theta s_\phi & c_\theta c_\phi \end{bmatrix} v_1 \tag{3.38}$$

Euler rates in terms of the body angular velocities can be written as

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & s_\phi t_\theta & c_\phi t_\theta \\ 0 & c_\phi & -s_\phi \\ 0 & s_\phi / c_\theta & c_\phi / c_\theta \end{bmatrix} v_2 \tag{3.39}$$

where velocities are described in a body fixed frame with liner velocity $v_1=[u \quad v \quad w]^T$ and angular velocity $v_2=[p \quad q \quad r]^T$.. Furthermore, following notations are used: *s.≡ sin (.), c.≡ cos (.), t.≡ tan (.).*

3.4.2 Dynamic Equations of Aircraft

The dynamic model of an aircraft is commonly described by 6 DOF equations which are derived from the *X, Y* and *Z* forces and *L, M* and *N* moment equations (See Fig. 3.28). The equations of motion for an airplane usually are written in a body-fixed coordinate system [103-105]. An aircraft center of mass is chosen as the origin for this system, and the orientation of the (right-handed) system of coordinate axes is chosen by convention which is illustrated in Fig.3.28.

- The x-axis lies in the symmetry plane of the vehicle and points forward;
- The z-axis lies in the symmetry plane of the vehicle, is perpendicular to the x-axis, and points down;
- The y-axis is perpendicular to the symmetry plane of the vehicle and points out the right wing.

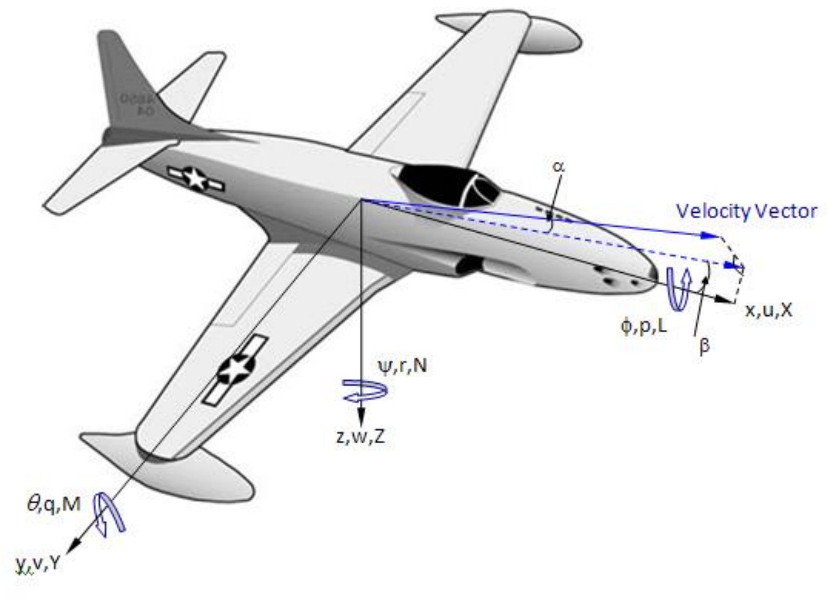


Figure 3.28: Body axis system with origin at center of gravity of a aircraft

The dynamic equations of motion are obtained from Newton's second law. The forces in the body *x*, *y* and *z* axis are given by:

$$X = \bar{q}SC_X$$
$$Y = \bar{q}SC_Y$$
$$Z = \bar{q}SC_Z \tag{3.40}$$

where $\bar{q}$ (Pa) is the dynamic pressure, *S* is the wing area, C_X , C_Y , and C_Z are the aerodynamic force coefficients in the *x*, *y* and *z* axis, respectively.

The summation of the forces in body *x*, *y* and *z* axis gives linear velocity equations:

$$\dot{u} = rv - qw + \frac{\bar{q}SC_X}{m} - gS_\theta + \frac{T}{m}$$
$$\dot{v} = pw - ru + \frac{\bar{q}SC_Y}{m} - gC_\theta S_\phi$$
$$\dot{w} = qu - pv + \frac{\bar{q}SC_Z}{m} - gC_\theta C_\phi \tag{3.41}$$

The moments about the body *x*, *y*, and *z* axis are given by:

$$L = \bar{q}Sbc_l$$
$$M = \bar{q}S\bar{c}c_m \tag{3.42}$$
$$N = \bar{q}Sbc_n$$

where c_l , c_m , and c_n are the non-dimensional moment coefficients, b is the wing span, and $\bar{c}$ is the wing mean aerodynamic chord.

Taking a moment about the aerodynamics center of the aircraft, the angular rate equations are given by:

$$\dot{p} - \frac{I_{xz}}{I_{xx}}\dot{r} = \frac{\bar{q}Sbc_l}{I_{xx}} - \frac{I_{zz} - I_{yy}}{I_{xx}}qr + \frac{I_{xz}}{I_{xx}}qp$$

$$\dot{q} = \frac{\bar{q}S\bar{c}c_m}{I_{yy}} - \frac{I_{xx} - I_{zz}}{I_{yy}}pr - \frac{I_{xz}}{I_{yy}}(p^2 - r^2) + \frac{I_p}{I_{yy}}\omega_p r$$

$$\dot{r} - \frac{I_{xz}}{I_{zz}}\dot{p} = \frac{\bar{q}Sbc_n}{I_{zz}} - \frac{I_{yy} - I_{xx}}{I_{zz}}pq - \frac{I_{xz}}{I_{zz}}qr - \frac{I_p}{I_{zz}}\omega_p q \tag{3.43}$$

where I_{xx} , I_{yy} , and I_{zz} are the moments of inertia of the body axis system. I_{xz} is the x-z body axis product of inertia. Since aircraft are symmetric with respect to the XZ plane, then I_{xy} and $I_{yz}$are both zero. I_p and ω_p are the moment of inertia for the population system and the propeller rotation speed, respectively.

The linearized small-disturbance longitudinal and lateral rigid body equation of motion for a fixed wing aircraft can be written as follows [103-105]:

$$\begin{gathered}
\dot{u} = X_u u + X_w w + X_q q + X_\theta \theta + X_{\delta_e}\delta_e + X_t \delta_\tau \\
\dot{w} = Z_u u + Z_W w + Z_q q + Z_\theta \theta + Z_{\delta_e}\delta_e + Z_t \delta_\tau \\
\dot{q} = M_u u + M_w w + M_q q + M_\theta \theta + M_{\delta_e}\delta_e + M_t \delta_\tau \\
\dot{v} = Y_v v + Y_p p - (u_0 - Y_r) r + Y_\phi \phi + Y_\psi \psi + Y_{\delta_a}\delta_a + Y_{\delta_r}\delta_r \\
\dot{p} = L_v v + L_p p + L_r r + L_{\delta_a}\delta_a + L_{\delta_r}\delta_r \\
\dot{r} = N_v v + N_p p + N_r r + N_{\delta_a}\delta_a + N_{\delta_r}\delta_r
\end{gathered} \tag{3.44}$$

where $\delta_{con} = [\boldsymbol{\delta}_e\ ,\ \boldsymbol{\delta}_a\ ,\ \boldsymbol{\delta}_r\ , \boldsymbol{\delta}_\tau]^T$ are control inputs, corresponding to the elevator, aileron, rudder deflection angles, and thrust, respectively. The stability and control derivatives used in this dynamic model are derived from a nonlinear UAV model using linearization. Therefore, these derivatives depend on the physical parameters and aerodynamic coefficients of the UAV. Several derivatives are of particular interest in this study and used as uncertainty parameters, which are X_u , X_w , Z_u , Z_w , X_t , $Z_{\delta e}$, Y_v , $Y_{\delta a}$, $Y_{\delta r}$ being inversely proportional to the aircraft mass *(m)*, X_t and $M_{\delta e}$ proportional to aerodynamic coefficients of $C_{\delta t}$ and $C_{m\delta e}$ respectively.

3.4.3 Motion Control and Obstacle Avoidance

Fig.3.29 provides an overview of the motion planning and control architecture. The goal of motion planning is to generate a desired trajectory so that the UAV can track. Aircraft longitudinal and lateral dynamics and kinematic equations are considered for the clearance process. The high level mission planer usually supplies waypoint information to the motion controller. Then the motion controller retrieves the waypoints and generates a desired trajectory. The inner-loop control law is responsible to compute the input signals that drive the motors and control surfaces to force the UAV to fly at a desired linear velocity and attitude so the collision avoidance path generated by the motion controller can be followed.

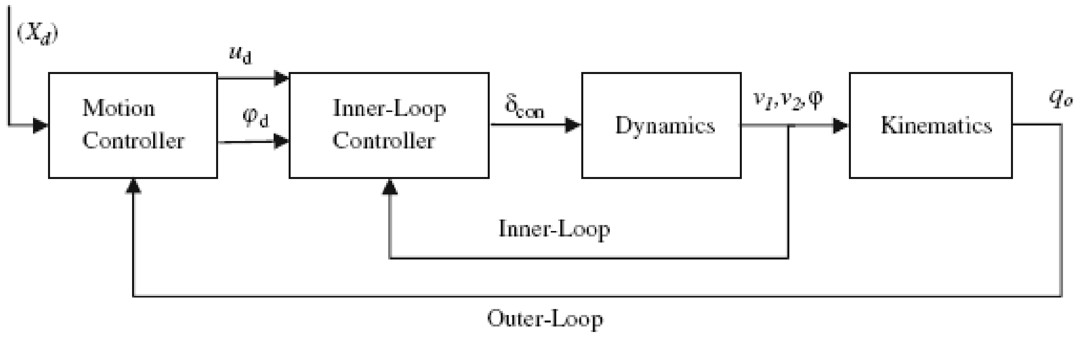


Figure 3.29: Motion planning and control structure

Obstacle Avoidance Algorithm

Due to the absence of a pilot, the use of UAVs has become increasingly popular in military and civilian applications. Path planning of UAVs with known and unknown obstacles is considered as one of the key enabling technologies in unmanned vehicle systems.An UAV has to find a collision-free path between the starting point and the goal (e.g. waypoint) in an environment containing various static obstacles. Specifically, spherical obstacles are considered in this study but it is applicable to other obstacles. In order to maintain safety, as shown in Fig.3.30, the influence range of an obstacle is determined from the radius of the obstacle plus a specified safe margin.

For a spherical obstacle, the influence range is chosen as the radius of r_{infl} which is greater than the radius of the obstacle (r_0) and the safe margin (r_{safe}). Letting $r_n = r_0 + r_{safe}$, the anti-collision condition is defined as $d_{min}>r_n$.

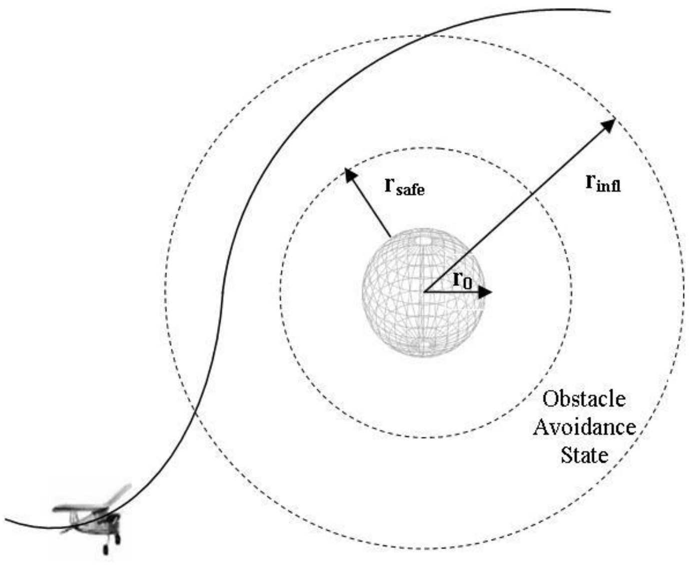


Figure 3.30: Obstacle avoidance clearance criterion

Potential field method is chosen as the path planning and the obstacle avoidance technique in 3D environment. In case study-1, the attractive and repulsive potential forces are discussed in 2D environment. Recall the Eqs. (3.11) and (3.13) to get the $F_{att}(q_o)$ and $F_{repi}(q_o)$ in 3D environment, where $q_o=[x, y, z]^T$ denotes the UAV current position in airspace. Therefore, the desired global velocities can be obtained and written as follows:

$$\begin{bmatrix}\dot{x}_d \\ \dot{y}_d \\ \dot{z}_d\end{bmatrix} = -\nabla\left(U_{att}(q_o) + U_{rep_i}(q_o)\right) = (F_{att}(q_o) + F_{rep_i}(q_o) \qquad (3.45)$$

After the desired global velocity is calculated by the potential field method, the corresponding desired linear velocity u_d and attitude $\boldsymbol{\varphi}_d = [\phi_d, \theta_d, \psi_d]^T$ can also be obtained based on UAV's kinematic model using the following equations:

$$u_d = k_u(\sqrt{\dot{x}_d^2 + \dot{y}_d^2 + \dot{z}_d^2}) \tag{3.46}$$

$$\phi_d = 0 \tag{3.47}$$

$$\theta_d = \ atan2(-\dot{z}_d\,, \sqrt{\dot{x}_d^2 + \dot{y}_d^2}\ ) \tag{3.48}$$

$$\psi_d = atan2(\dot{y}_d\,, \dot{x}_d) \tag{3.49}$$

where gain k_u is introduced to allow for additional freedom in weighting the velocity commands. The pitch and yaw angle guidance laws are designed so that the vehicle's longitudinal axis steers to align with the gradient of the potential field. The roll angle guidance law is designed to maintain the level flight.

Inner-Loop Controller

To accomplish the goal of driving the UAV flying at the desired linear velocity u_d and desired attitude angles $\boldsymbol{\varphi}_d$, the first step is to compute the error between the true linear, the attitude angles and the desired ones, respectively. To this effect, let $e_u = (u_d - u)$, $e_{\delta a} = (\phi_d - \phi)$, $e_{\delta e} = (\theta_d - \theta)$ and $e_{\delta r} = (\psi_d - \psi)$ denote the linear velocity and attitude angle errors, respectively. A simple PID control law is proposed as

Elevator control signal:

$$\delta_e = K_{p1} e_{\delta_e} + K_{i1} \int_o^t e_{\delta_e}(\tau) d\tau + K_{d1} \frac{de_{\delta_e}}{dt}$$

Aileron control signal:

$$\delta_a = K_{p2} e_{\delta_a} + K_{i2} \int_o^t e_{\delta_a}(\tau) d\tau + K_{d2} \frac{de_{\delta_a}}{dt}$$

Rudder control signal:

$$\delta_r = K_{p3} e_{\delta_r} + K_{i3} \int_o^t e_{\delta_r}(\tau) d\tau + K_{d3} \frac{de_{\delta_r}}{dt}$$

Throttle control signal:

$$\delta_\tau = K_{p4} e_u + K_{i4} \int_o^t e_u(\tau) d\tau + K_{d4} \frac{de_u}{dt} \tag{3.50}$$

Four PID controllers are designed for controlling linear velocity and three attitude angles. As the angular rates *p*, *q* and r are available in flight control, the corresponding derivative terms in the PID controllers are replaced by their angular rate feedback.

3.4.4 Collision Avoidance at Nominal Parameters

In this section, the proposed collision avoidance algorithm and controller are validated at the nominal parameters. The simulation results for a UAV approaching a spherical obstacle are presented at the nominal parameters. The nominal parameter values are $m=1.9$ *kg*, $C_{m\delta e}$ =-*1.13*, and $C_{\delta t}$ =*12.19*. The initial linear and angular velocity vectors are (*15, 0, 0) m/s* and *(0, 0,0) rad/s* for the nominal case. The initial Euler angle is *(0, 0, 0.9) rad*. Safe margin is chosen as 5*m*. The PID controller gains and motion planner parameters for potential field force are also tuned and set to fixed values for the verification process. In the simulation, the initial departure point is *(0, 0, 20)m*, and the spherical obstacle is located at *(250, 250, -10) m* with a radius r_0 of *20m.* Therefore, the safety radius is *25m* including safe margin. The simulation result at the nominal parameters is shown in Fig.3.31.

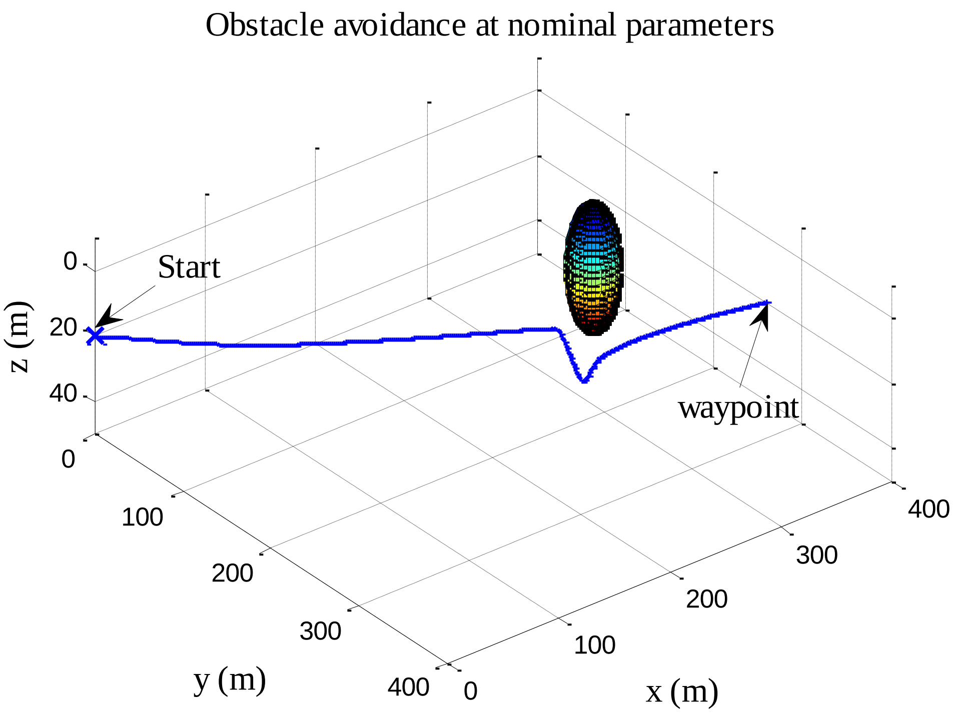


Figure 3.31: Simulation result for UAV collision avoidance at nominal parameters

The minimum distance to the obstacle is obtained as *29.6166m* which is greater than obstacle safety radius *25m* *($d_{min}>r_n$)*. This concludes that the obstacle avoidance algorithm works correctly at the nominal parameters.

3.4.5 Initial Robustness Analysis

Initial robustness analysis of the proposed algorithm is carried out in this section. Uncertainties are considered in the dynamic model (mass and two aerodynamic coefficients), and each uncertain parameter is allowed to vary within ± *(10 or 20)%* of its nominal value. These are firstly considered within lower and upper bounds, i.e. m= [1.52 , 2.28] *kg*, $C_{m\delta e}$ = [-1.243 , -1.017] and $C_{\delta t}$= [10.971 , 13.409]. For the purpose of comparison, the uncertain parameters are normalized to have a variation within the range. Fig.3.32 shows variations of the minimum distance to the obstacle with respect to the normalized uncertain parameters of mass, $C_{m\delta e}$ and $C_{\delta t}$. There is a significant variation in the distance with the variations of these uncertain parameters. The minimum distance to the obstacle monotonically decreases with the increase of the m and $C_{m\delta e}$, while d_{min} increases with the increase of $C_{\delta t}$.

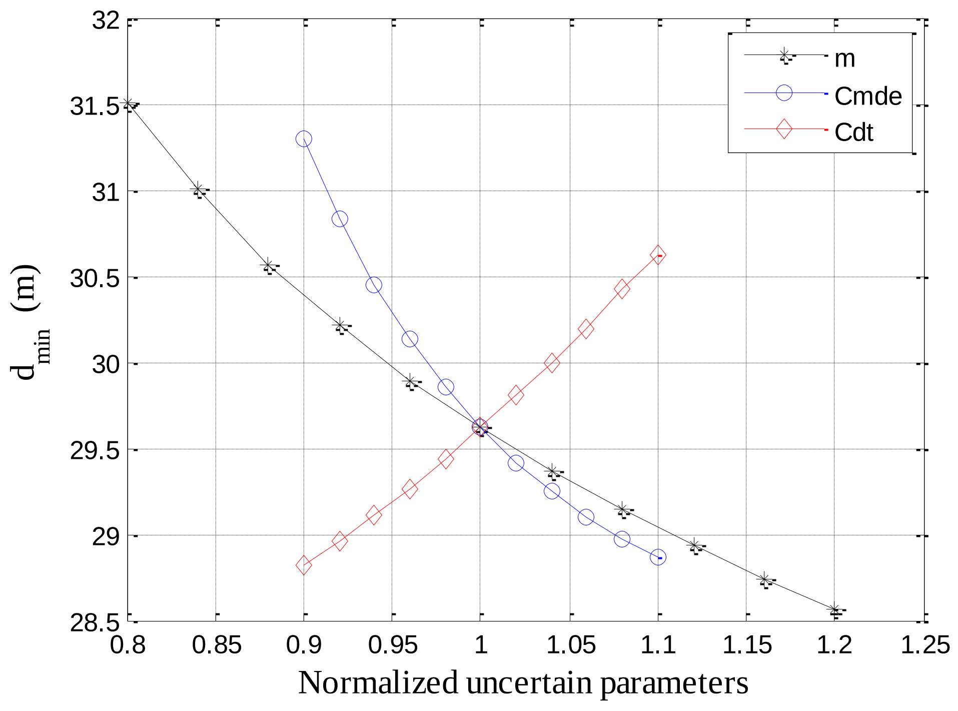


Figure 3.32: Mass, $C_{m\delta e}$ and $C_{\delta t}$ variations

3.5 Case Study- 4

In this section, robustness analysis of moving OAS is considered for UAVs. Kinematic and dynamic model equations of UAV which are described in the previous section of 3.4.1 are used for this study. Moving obstacle avoidance algorithm using with potential field method is chosen and applied to the UAVs to avoid collisions. Moving obstacle avoidance algorithm in 2D is discussed in the section 3.3.2 and 3.3.4, and this algorithm is extended to the 3D environment for UAVs OAS. The safety radius including safe margin is chosen as *50m*. The simulation results at nominal parameters are shown in Fig.3.33. The minimum distance to the obstacle is obtained as 59.23mwhich is greater than the safety radius (*dmin> r*). Therefore, the moving obstacle avoidance algorithm functions correctly at nominal parameters. Initial robustness analysis for moving OAS is also studied. Each uncertain parameter is allowed to vary within ± *20%* of its nominal value. Fig.3.34 shows variations of the minimum distance to the obstacle with respect to the normalized uncertain parameters of mass, $C_{m\delta e}$ and $C_{\delta t}$. The minimum distance to the obstacle increases with the increase of the *m* and $C_{m\delta e,}$ and $C_{\delta t}$.

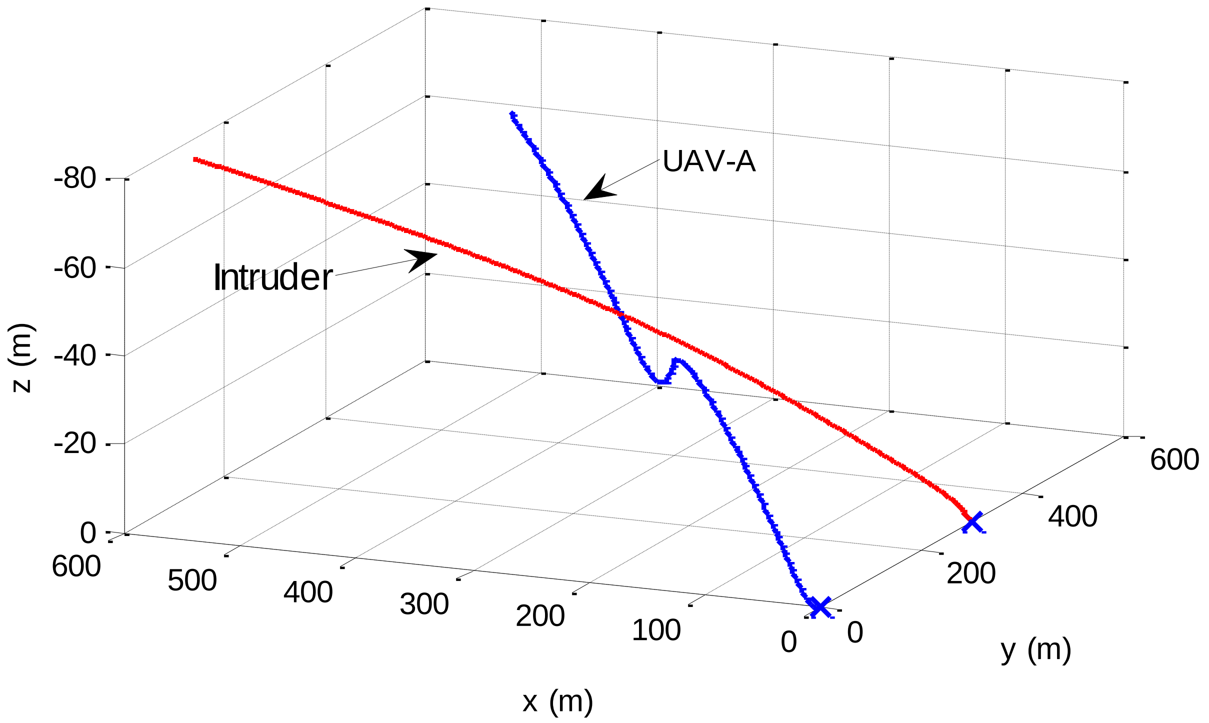


Figure 3.33: Simulation result for UAVs collision avoidance at nominal parameters

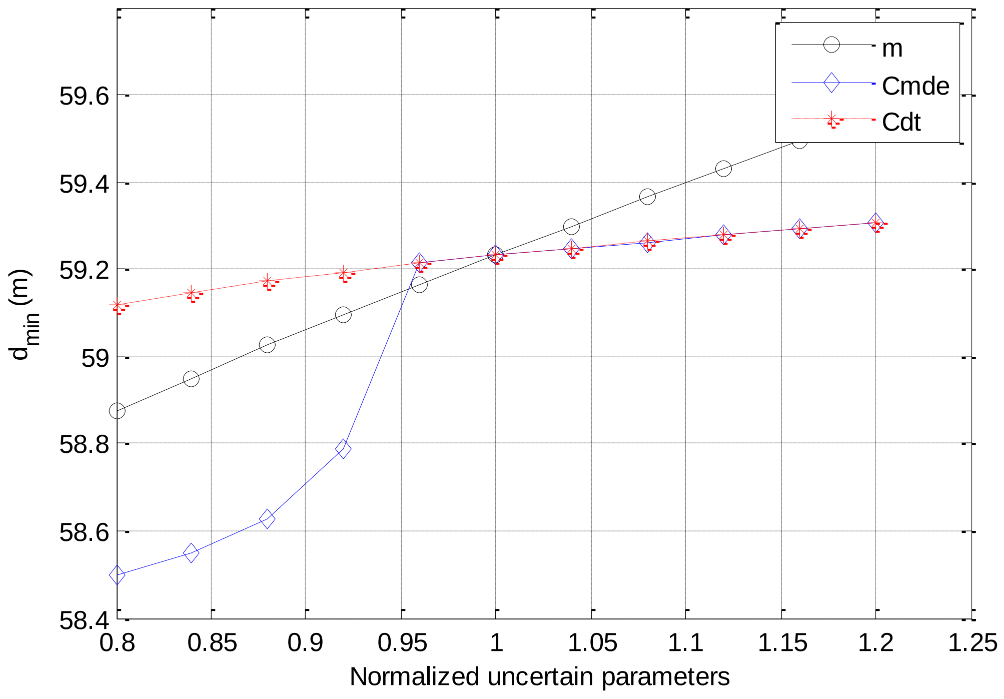


Figure 3.34: Mass, $C_{m\delta e}$ and $C_{\delta t}$ variations in moving OAS

3.6 Conclusion

In this chapter, the uncertainty analysis of static and moving OAS for unmanned vehicles is discussed in details. In order to apply the verification process of OAS for unmanned vehicles which is described in Chapter 1, there are four case studies considered. As a benchmark, kinematic and dynamic equations of the unmanned vehicles are introduced and the controller is chosen based on these equations. The inner-outer-loop control architecture is used where the inner-loop controller is a PID speed controller. A local planner in the outer-loop is developed using the potential field method for static and moving obstacles. It is necessary for demonstrating the mismatching between the model used in algorithm development and the vehicle, and for presenting the proposed verification process. After analysing the OAS at nominal case, it is extended to the robustness analysis. The uncertainty analysis results clearly show that for different uncertain parameters, the influence on the minimum distance to obstacle could be quite different. Worst-case analysis is the next step of the verification process. In the next chapters, the local and global optimisation algorithms will be applied to find the worst-case conditions and worst-case parameters.

Chapter 4

Local Optimisation-based worst-case Analysis for Obstacle Avoidance Systems

4.1. Introduction

Optimisation algorithms are becoming increasingly popular in engineering design applications. It is expected that the design solution obtained through an optimisation procedure is better than other solutions in terms of the chosen objective- such as cost, efficiency, safety, etc. Real engineering design decisions at the system level typically involve both technical (e.g. maximise performance) and economic (e.g. minimise cost) considerations. The goal of all decisions is either to minimise the effort required or to maximize the desired benefit. The effort required or the benefit desired in any practical situation can be expressed as a function of certain decision variables. Therefore, the cost functions depend on a set of input parameters and constraints. Nonlinear optimisation problems occur naturally and frequently in the various field including chemistry, physics, economics, engineering and mathematics, and there is a need for provably convergent, implementable algorithms to solve such problems.

The verification of collision avoidance systems can be stated as a robustness analysis problem, where a suitably defined anti-collision condition must be checked within the most significant variations of vehicles parameters. In order to find the worst-case parameters and worst-case condition, the efficient method of optimisation-based verification algorithm is applied to the OAS. This optimisation-based verification method can be applied to linear and nonlinear robustness analysis and also to different static and moving obstacle avoidance algorithms. Therefore, it is a very flexible and efficient method for the robustness analysis of collision avoidance system.

In the last chapter, four case studies were considered for OAS verification, and uncertainty analysis were investigated. In this chapter, local optimisation-based worst case analysis is studied for four case studies. The local optimisation algorithm is applied to the OAS to find the combination of parametric uncertainties that gives the worst violation of the criterion defined in Eq.1.1. Nonlinear optimisation problem arising in OAS verification often have multiple local optima and expensive function evaluations. Therefore, the issue of whether to use local or global optimisation, and the associated impact on computation time and guarantee the global minima are key consideration for this V&V problem.

4.2 Optimisation

Optimisation can be defined as the process of finding the conditions that give the maximum or minimum value of a function. In Fig.4.1, the point X_{min} corresponds to the minimum value of the function *f(X).*

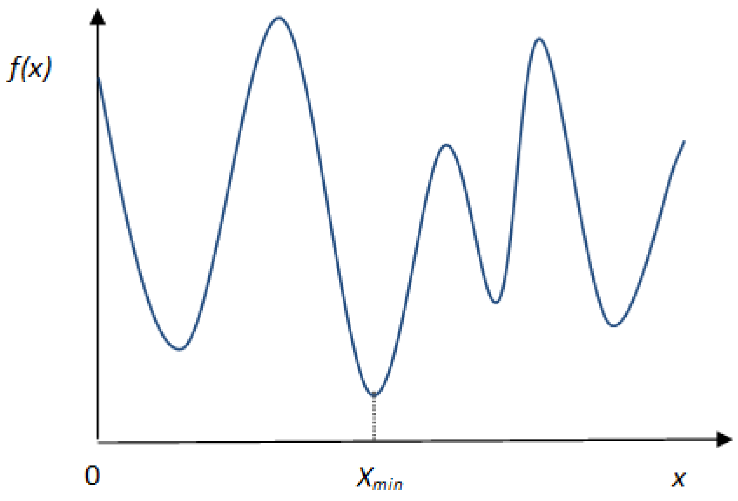


Figure 4.1: Minimum of *f*(X)

Since an optimisation algorithm requires comparison of a number of design solutions, it is usually time consuming and computationally expensive. The optimisation algorithms used to solve the problems depends on the type of the objective function, design variables and constraints. The optimisation problem can be mathematically written in a special format, known as nonlinear programming (NLP) format. Denoting the design variables is a column vector X=$[x_1, x_2, ...,x_n]^T$, the objective function is a scalar quantity of $f(X)$.

$$\begin{aligned} &\textit{Minimize} && f(X) \\ &\textit{Subject to} && h(X)=0 \\ & && g(X)\leq 0 \\ & && X_L\leq X \leq X_U \end{aligned} \tag{4.1}$$

where	$f(X)$	is a real value function called the objective function
	X	is a vector of n real independent variables called decision variables
	h	represents the set of equality constraint functions of X
	g	represents the set of inequality constraint function of X
	X_L , X_U	represent the lower and upper bounds on feasible values of X

Eq.(4.1) represents the general form of a constrained objective optimisation problem. Most of practical problems are often constrained by a number of restrictions imposed on the decision variables. While proposed as a minimisation problem, it could easily be converted into a maximisation problem by taking the negative of $f(X)$, such that minimising $-f$ is equivalent to maximising f without loss of generality [65; 81; 82]. Therefore, every optimisation problem must be formulated in the specific format.

The objective in a design problem and the associated design parameters vary from product to product. Therefore, different optimisation techniques need to be used in different problems. An optimisation algorithm accepts an optimisation problem in a particular format. The designers need to choose the correct optimisation format, design variables, constraints, objective function and variable bounds. A design problem usually involves many design parameters, of which some are more significant to the proper working of the design. These parameters are called design variables in the optimisation procedures. There is no proper guideline to choose the parameters which may be important in a problem, because one parameter may be more important with respect to minimizing the overall cost of the design, while it may be insignificant with respect to maximizing the performance of the design product. The choice of the important parameters in an optimisation problem depends on the problem. However, it is important to understand that the efficiency and speed of optimisation algorithms depend on the number of chosen design variables. Another important task is to identify the constraints associated with the optimisation problem. The constraints represent some functional relationships among the design variables and other design parameters satisfying certain physical limitations. There is no unique way to formulate a constraint in all problems, again number of constraints to be included in the optimisation problem depend on the problem. There are usually two types of constraints which are of an inequality type or of an equality type. Inequality constraints state that the functional relationships among design variables are either greater than, smaller than, or equal to, a resource value. Equality constraints state that the functional relationships should exactly match a resource value. In the optimisation formulation procedure, the objective function is defined in terms of the design variables and constraints. The objective function can be of two types: either the objective function is to be maximised or it has to be minimised. As mentioned earlier, the duality principle helps by allowing the same algorithm to be used for minimization or maximization with a minor change in the objective function instead of a change in the entire algorithm. If the algorithm is developed for solving a minimization problem, it can also be used to solve a maximization problem by simply multiplying the objective function by -1 and vice versa [81; 82].

Another task of the optimisation formulation is to set the minimum and the maximum bounds on each design variables. In these problems, the constraints completely surround the feasible region. In general, all *n* variables are restricted to lie within the lower and upper bounds. Bound-constrained optimisation problem is an important role in real applications because parameters that describe physical quantities are often constrained to lie in a given range [81, 82].

After the above tasks are completed, the optimisation problem can be mathematically written in a special format known as nonlinear programming (NLP) format which is described in Eq.(4.1). For three decades, many mathematical programming (Constrained optimisation) algorithms has been developed to solve optimisation problem. Most mathematical optimisation applications are focused and developed for continuous variables such that in bound constraints. Therefore, handling constraints in control system design is an important part in real world problems [83; 84]. The purpose of the optimisation is to find the value of the parameters in order to minimise or maximise the output of the objective function.

Depending on the type of variables, constraints, and objectives, there are several methods that can be used for optimisation. A search method for solving continuous constraint problems can be divided into two categories which are *global search* and *local search*. Fig.4.2 which is generated by *peaks* function of MATLAB illustrates the local and global optima.

Global search methods spend a great amount of effort exploring the global search space, whereas local search methods focus on converging to local optimal solutions. The search space or design space can be convex or non-convex. If the search space is convex, both local and global optimisation algorithms will converge to the true global solution. In the case of non-convex search space, the local optimisation algorithms provide a local solution rather than the true global solution.

A gradient-based local optimisation method is described in this Chapter. It is called as a *"Sequential Quadratic Programming-(SQP)"* which is designed as *"fmincon"* function in *MATLAB* optimisation toolbox [86].

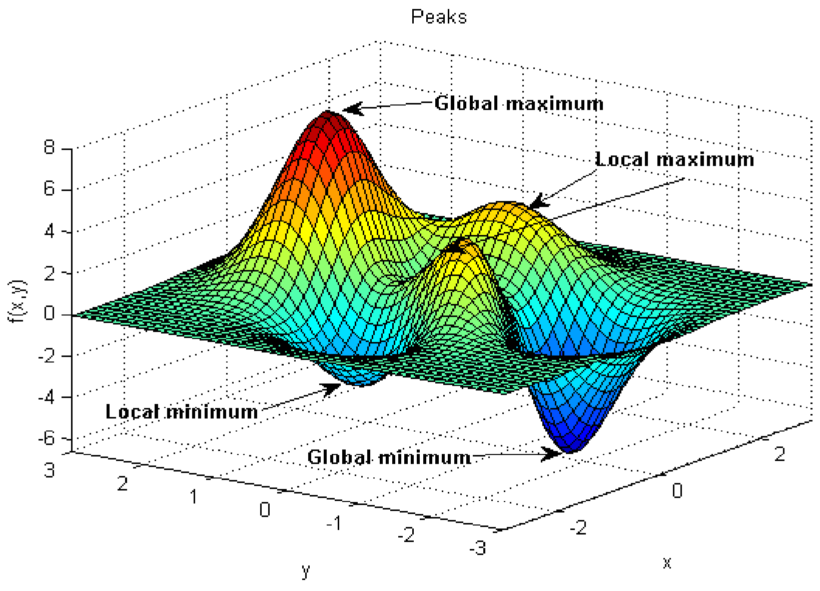


Figure 4.2: An example illustrating local and global optima

4.3 Local Optimisation Method

Starting from initial solution, local optimisation algorithms try to find the nearest local optimal solution. Thus, determining a good initial solution often becomes very critical to obtain a satisfactory optimisation result. In the entire solution space optimisation process, local optimisation algorithm tends to be trapped in local optima depending on the initial solution. Local optimisation is widely applicable, since they only require differentiability of the objective function and constraint functions. It can rapidly find a set of parameter values and local minima, but not guaranteed to be the absolute worst possible.

Sequential Quadratic Programming (SQP):

SQP method is one of the most powerful optimisation algorithms for solving medium-size nonlinear constrained optimisation problems when the functions and gradients can be evaluated with high precision. It is an iterative method starting from an initial point and converging to a local minimum. It is a standard general purpose algorithm for solving smooth and well-scaled nonlinear optimisation problems. They generally require few iterations and function evaluations. In many situations, the local gradients will not be available analytically. In all such situations numerical approximations of gradients have to be computed and this might cause slower and less reliable performance, especially when the function evaluations are noisy. SQP is implemented in MATLAB’s *fmincon* function, which handles equality and inequality constraints [85, 86, 87].

4.4 Verification Benchmark- Local Optimisation Results

The local optimisation algorithm *fmincon* is first considered for the verification of OAS. A local method converges to whether a local or global optimum entirely depends on the given starting points in the search space. However, in verification of OAS problems only very little information is available as to where to start the optimisation. Because of the number of uncertain parameters and nonlinearity of the system, it is very difficult to choose the initial values of the uncertain parameters. The function *fmincon* is applied with different starting points to the problem of evaluating a worst-case condition for the unmanned vehicle OAS which are described in chapter 3. A medium-scale SQP method is chosen for the optimisation. The *fmincon* finds the minimum of a scalar function, which can be multivariable, subject to given constraints. Constraints are chosen within the lower and upper bounds of the uncertainty in the parameters. This iteration is repeated until a specified termination criterion (either maximum number of function evaluations or convergence accuracy) is met.

To assess the safety of vehicles, the minimum distance to the obstacle *(d_{min})* is defined as the objective function in the time domain.

$$\begin{gathered} d_{min} = min(d(t)) \quad \text{for} \quad t \leq T\ (sec) \\ s.t \quad P_L \leq P \leq P_U \end{gathered} \tag{4.2}$$

where P is the uncertain parameters set; P_L and P_U are lower and upper bounds of P; T is the time period of the collision avoidance manoeuvre; *d(t)* is the distance to the obstacle and is calculated using simulation with the completed model of the vehicle in Fig 1.5.

4.4.1 Case Study-1: Simple Unicycle

The results of the minimum distance to the obstacle and worst case parameters with different starting points are given in Table.4.1. The uncertain parameters of mass and inertia are chosen within the lower and upper bound which were discussed in chapter 3. The 3D plots of *fmincon* optimisation results at case 1 & 2 are shown in Figs.4.3- 4.4. It is shown that even for a simple case study where only mass and inertia variations are considered, a local optimisation based verification method may fail to identify the worst case. The reason of why the *fmincon* does not converge to the global minimum is because it is a non-convex problem and it has many local minima, each local minima not a global one.

TABLE.4.1 LOCAL OPTIMISATION RESULTS FOR SIMPLE UNICYCLE

fmincon	Starting point	[m (kg) , J(kgm^2)]	d_{min} *(m)*
Case-1	**[4.5, 0.05]**	**[4.4224, 0.06]**	**0.9394**
Case-2	[5.5, 0.048]	[5.1925, 0.06]	0.9382

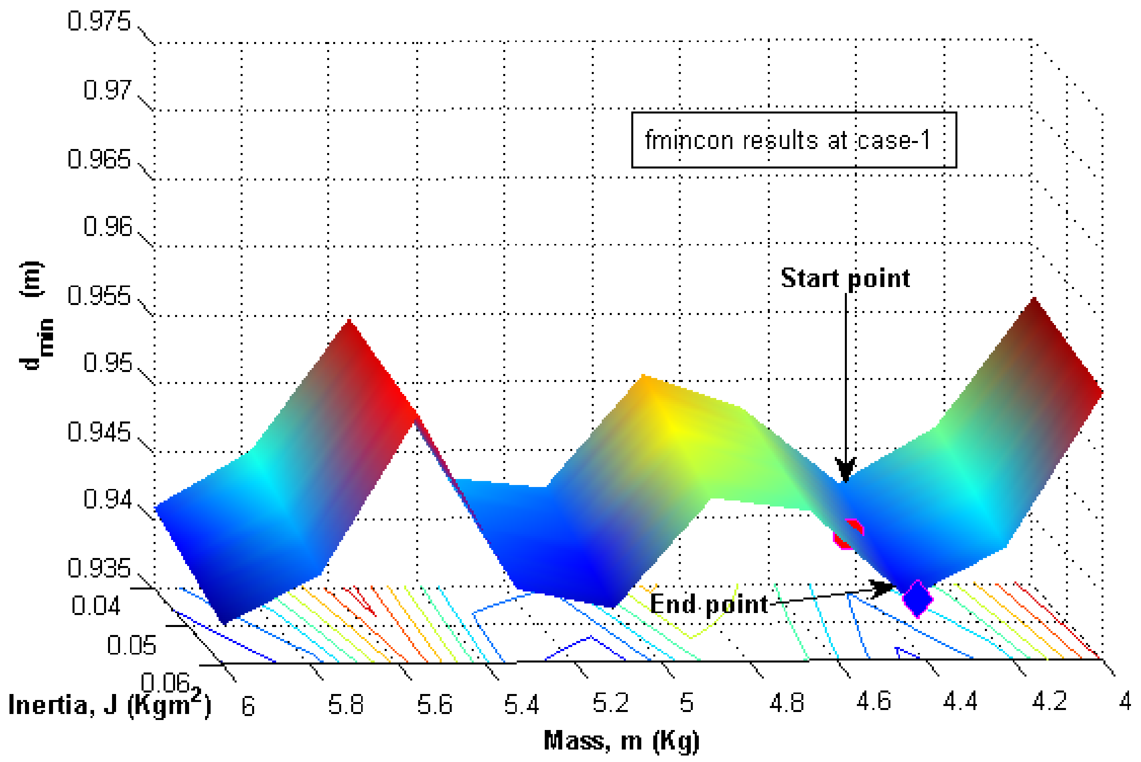


Figure 4.3: *fmincon* results at case-1

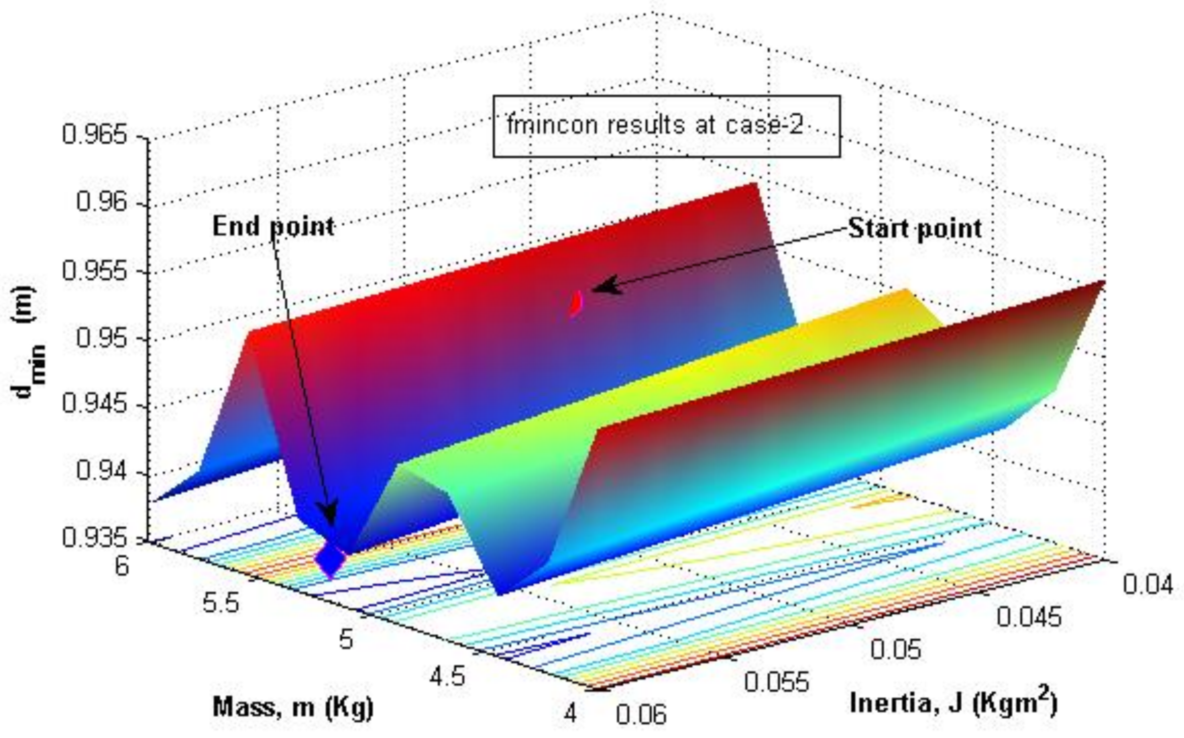


Figure 4.4: *fmincon* results at case-2

4.4.2 Case Study-2: Pioneer 3-DX

The local optimisation method is applied with different starting points to the problem of evaluating a clearance criterion for the moving OAS of Pioneer 3-DX. Eight most significant parameters are chosen for the robustness analysis. Lower and upper bounds of parameters are given in Table.3.2 to determine the worst-case parameters. The *fmincon* tries to find iteratively a minimum at an initial estimate. Therefore, different starting points are specified and compared the results.

In Table.4.2, the results of the minimum distance to the obstacle and worst-case parameters with different starting points are given. At case-1, it converges to the minimum distance of 6.8164m while it is 5.8722m at case-2. And also, it can be seen that there are huge differences in the converging parameters set. Therefore, the results clearly show that *fmincon* does not give the same solutions with the different starting points because a local optimisation solution quality depends heavily on the initial points picked. Local optimization-based method is not suitable for this case study. Moreover, Fig.4.5 shows the worst-case violation in distance to the obstacle at case-1 & 2 and at nominal parameters.

TABLE.4.2. LOCAL OPTIMIZATION RESULTS FOR PIONEER 3-DX ROBOT

Algorithm	Starting point $[m, B_e, \delta_u, \delta_\omega, I_e, \bar{u}^s, \Delta x, \Delta y]$	Convergent point $[m, B_e, \delta_u, \delta_\omega, I_e, \bar{u}^s, \Delta x, \Delta y]$	d_{min}(m)
fmincon-case 1	[20, 1.0, 0.2, 0.2, 0.6, 0.03, 0, 0]	[18.271, 0.48, 0.9, 0.1, 0.204, 0.02, 0.5, - 0.5]	6.8164
fmincon-case2	**[30, 1.0, 0.8, 0.8, 3.0, 0.07, 0.4, 0.4]**	**[34, 0.48, 0.9, 0.1 , 3.8, 0.02, 0.5, - 0.5]**	**5.8722**

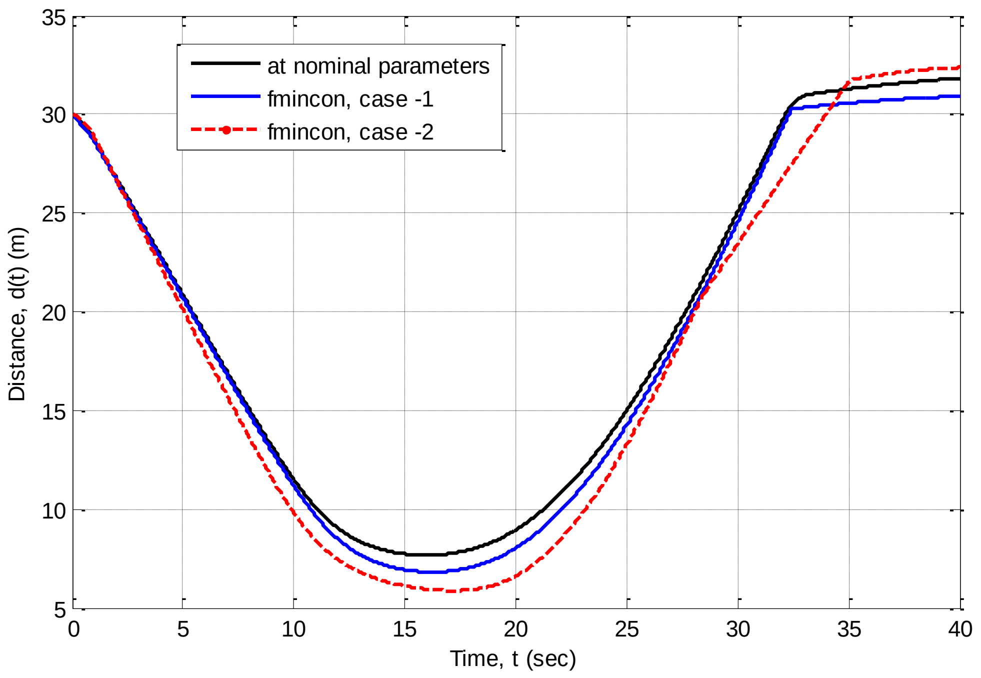


Figure 4.5: *fmincon* worst-case violation in distance to the obstacle

4.4.3 Case Study-3: UAV model-Static OAS

The worst parameter combination and minimum distance to the obstacle found by *fmincon* is shown in Table.4.3. The nonlinear optimisation problem considered in this case study is likely to have multiple local optima. Optimisation algorithm is run starting from a different randomly chosen initial guess for the uncertain parameters. The results clearly show that *fmincon* does not give the same solutions for this problem because the solution for a local optimisation algorithm depends on the starting point. Therefore, global optimisation methods are studied to find the true worst-case in the next chapter. Velocity during the UAV manoeuvre response at nominal and worst-case parameters is shown in Fig.4.6.

TABLE.4.3. LOCAL OPTIMISATION RESULTS FOR UAV STATIC OAS

Algorithm	Starting point $[m, C_{m\delta e}, C_{\delta t}]$	Convergent point $[m, C_{m\delta e}, C_{\delta t}]$	$d_{min}(m)$
fmincon	**[1.52, -1.13, 11.8243]**	**[2.28, -1.243, 10.971]**	**29.0235**
fmincon	[1.71, -1.13, 13.409]	[2.28, -1.0193, 10.971]	27.4967

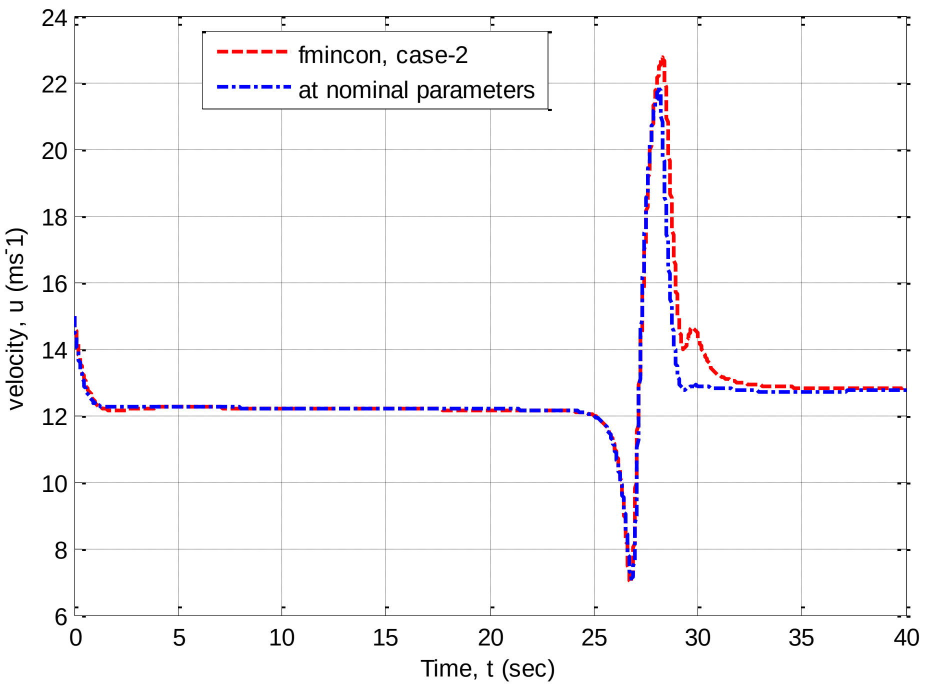


Figure 4.6: *fmincon* Worst-case violation in UAV velocity

4.4.4 Case Study-4: UAV model-Moving OAS

The local optimisation method is applied with different starting points to the problem of evaluating a clearance criterion for the moving OAS for UAVs. Three uncertain parameters are chosen for the robustness analysis. Lower and upper bounds of parameters are given to determine the worst-case parameters. The *fmincon* tries to find iteratively a minimum at an initial estimate. Therefore, different starting points are specified and compared the results.

In Table.4.4, the results of the minimum distance to the obstacle and worst-case parameters with different starting points are given. At nominal parameters, the minimum distance to the obstacle is obtained as 59.23m. At case-1, it converges to the minimum distance of 58.35m while it is 57.9m at case-2. Therefore, the results clearly show that *fmincon* does not give the same solutions with the different starting points. Local optimisation-based method is not suitable for this benchmark study. Because of this worst-case violation of the optimal solution, the global optimisation methods are considered to find the true worst-case. Fig.4.7 shows the worst case violation in velocity during the UAV manoeuvre.

TABLE.4.4. LOCAL OPTIMISATION RESULTS FOR UAV MOVING OAS

Algorithm	Starting point *[m, $C_{m\delta e}$, $C_{\delta t}$]*	Convergent point *[m, $C_{m\delta e}$, $C_{\delta t}$]*	*$d_{min}(m)$*
fmincon	[1.52, -1.13, 9.752]	[1.958, -1.356, 9.752]	58.35
fmincon	**[1.71, -1.13, 13.409]**	**[1.52, -1.356, 9.752]**	**57.90**

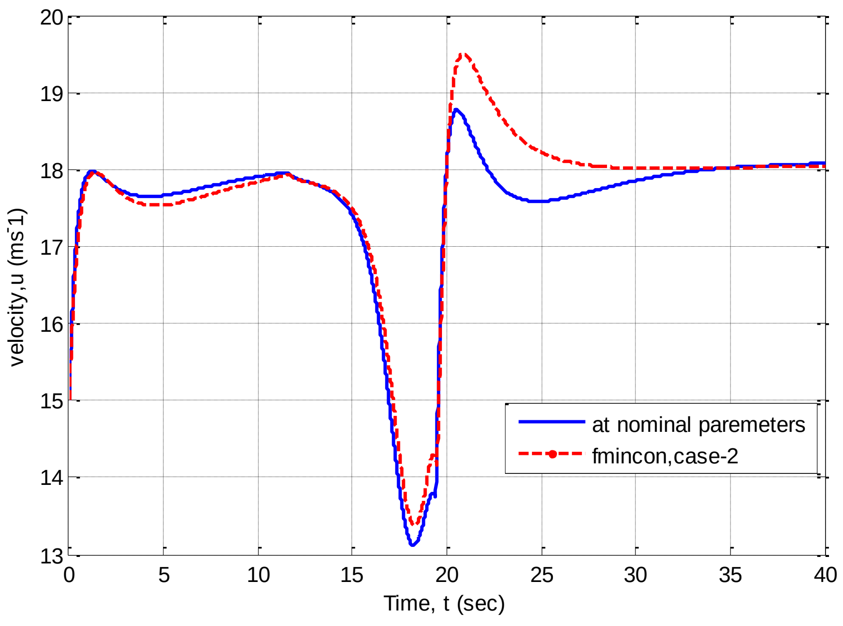


Figure 4.7: *fmincon* worst-case violation in UAV velocity

4.5 Conclusion

Safety is a paramount consideration in developing unmanned vehicles. In this chapter, the safety analysis of OAS is presented where optimisation-based methods have been developed for verification of obstacle avoidance algorithms. The key idea in this approach is that in optimisation, it is not necessary to evaluate a cost function over all possible solutions to find the optimal solution. However different from many optimisation problems, it is important to find all the possible worst cases in the worst case analysis of safety critical functions like obstacle avoidance. To demonstrate the challenges of the problem and the effectiveness of the proposed optimisation-based verification process, four benchmark studies are presented in the last chapter. An optimisation-based automatic search approach is proposed to find the worst cases and check whether the safety criterion is satisfied under all possible uncertainties. Minimum distance to the obstacle is defined as a clearance criterion.

In this chapter, the verification process of OAS is presented as a standard nonlinear programming problem with uncertain parameters bounds. Local optimisation method is applied to the four case studies to find the true worse case. Of the presented local method results for four case studies, it is difficult to select the best one, because the convergence proofs depend on the smoothness of the objective function. The results are clearly shown that all problem discussed in this chapter are non-convex problem and it has many local minima. It is demonstrated that local nonlinear optimisation method is not adequate for four case studies, and global optimisation technique is essential to find the true worse-case condition and worst-case parameters set which will be discussed in the next chapter.

Chapter 5

Global Optimisation-based worst-case Analysis for Obstacle Avoidance Systems

5.1. Introduction

Global optimisation is aimed at finding the true global optima solution of constrained optimisation problems which may have various local optima. Due to the nonlinear nature of the real systems, the optimisation problems are frequently non-convex. A non-linear optimisation problem is difficult to solve because the nonlinear constraints form feasible regions that are difficult to find, and also the nonlinear objective contain local minima that traps the search process. Nonlinear optimisation methods can be classified as local optimisation and global optimisation methods.

In last chapter, local optimisation method was applied to the four case studies, and results were shown that the local optimisation methods may fail to find the optimal solutions for all problems. Therefore, these methods may miss an unsafe point. To overcome local minima problem, global optimisation methods are applied to find the worst-case. Finding the global minimum of a nonlinear constrained optimisation problem is a challenging task. Generally, global optimisation algorithms are derived to find the globally optimal solutions. However, in many engineering applications, finding the global minima is a very time-consuming process due to its computational complexity. And also, guarantee the global optimal solution is a challenging task. The mechanism of escaping from local minima determines the efficiency of a global optimisation algorithm. Global optimisation methods can be classified as either stochastic or deterministic. Stochastic methods evaluate the objective function at randomly sampled points from the solution space. These stochastic global optimisation methods depend on probability conditions to make decisions. Therefore, these algorithms cannot guarantee the global optimal. On the other hand, the deterministic methods do not involve any elements of randomness, and these methods evaluate the objective function satisfies certain conditions, such as Lipschitz condition. Therefore, these algorithms can guarantee the optimal solution. Hybrid optimisations try to get the best optimum value of both methods, i.e. to combine global and local optimisation methods in order to reduce their weakness. These three general classes in global optimisation which are given below: [85 ; 93; 94; 95]

- Deterministic Optimisation
  - Grid search
  - Branch and Bound
  - Interval arithmetic methods
  - DIviding RECTangles (DIRECT)

- Stochastic Optimisation (Evolutionary global optimisation)
  - Simulated Annealing
  - Tabu search
  - Genetic Algorithm
  - Differential Evolution
  - Ant Colony Simulation
  - Particle Swarm Optimisation
  - GLOBAL algorithm

- Hybrid Optimisation
  - Global and local optimisation

Stochastic optimisation algorithms have been studied in the literature over the last three decades. Stochastic optimisation methods including GA and GLOBAL algorithms are considered to find the global minimum of the minimum distance to the obstacle for the four case studies. After that, the deterministic global algorithm of DIRECT (DIviding RECTangles) method is also applied to the OAS. The set of parameters is chosen within the bound range because these parameters are uncertain or may vary during operation.

5.2 Stochastic Global Optimisation

5.2.1 Genetic Algorithms

Genetic Algorithms (GA's) are general purpose stochastic search and optimisation algorithms, based on genetic and evolutionary principles. The theory and practice of the GA was originally invented by John Holland [73] in 1960s and was fully elaborated in his book 'Adaption in Natural and Artificial Systems' published in 1975. The basic idea of the approach is to start with a set of designs, randomly generated using the allowable values for each design variable. Each design is also assigned a fitness value. The process is continued until a stopping criterion is satisfied or the number of iterations exceeds as a specified limit. Three genetic operators are used to accomplish this task: Selection, Crossover, and Mutation [73 ; 96 ; 97] .

Selection is an operator where an old design is copied into the new population according to the design's fitness. In other words, selection is a process of selecting a set of designs from the current population and carrying them into next generation. There are many different strategies to implement this selection operator including roulette wheel selection, tournament selection and stochastic universal sampling. *Crossover* exchanges parts of solutions from two or more individuals, called parents, and combines these parts to generate new individuals, called children, with a crossover probability. There are a lot of ways to implement a crossover operator. The well-known crossover operators include one-point crossover. *Mutation* is the third step that safeguards the process from a complete premature loss of valuable genetic material during selection and crossover. It usually alters some pieces of individuals to form perturbed solutions. In contrast to crossover, which operates on two or more individuals, mutation operates on a single individual. One of the most popular mutation operators is the bitwise mutation, in which each bit in a binary string is complemented with a mutation probability. The foregoing three steps are repeated for successive generations of the population until no further improvement in fitness is attainable [85 ; 87 ; 96 ; 97 ; 99].

GAs, differing from other search techniques, start with an initial set of random solutions called population. Each individual in the population is called a chromosome, representing a solution to the problem. A chromosome is usually a string of symbols. The chromosomes evolve through successive iterations, called generations. During each generation, the chromosomes are evaluated, using some measures of fitness. To create the next generation, new chromosomes, called offspring, are formed by either (a) merging two chromosomes from current generation using a crossover operator or (b) modifying a chromosome using a mutation operator. A new generation is formed by (a) selecting, according to the fitness values, some of the parents and offspring and (b) rejecting others so as to keep the population size constant. After several generations, the algorithms converge to the best chromosome, which represents the optimum or suboptimal solution to the problem [98].

5.2.2 GLOBAL Algorithm

GLOBAL algorithm was developed by Csendes et al [101] in 1988. It is a modified version of the stochastic algorithm by Boender et al [100] implemented in FORTRAN. The new implementation GLOBAL.m has been written in MATLAB. It is a multistart clustering algorithm. It has two phases i.e. a global phase and a local phase. The global phase consists of sampling and clustering, while the local phase is based on local searches. A general clustering method starts with the generation of a uniform sample in the search space (the region defined by lower and upper bounds). After transforming the sample (by selecting a user set percentage of the sample points with the lowest function values), the clustering procedure is applied. Then, the local search is started from those points which have not been assigned to a cluster. GLOBAL uses the Single Linkage clustering rule [95, 101].

The new MATLAB based program is freely available for academic purposes at [101 ; 102]. It is the bound constrained global optimisation problems with black-box type objective function.

$$\begin{array}{l} \min f(x) \\ x \in X,\ X=\{a_i \le x_i \le b_i;\ i=1 \text{ to } n\} \end{array} \qquad (5.1)$$

where $f : R^n \rightarrow R$ is a real valued function, X is the feasibility, an n-dimensional interval with vectors of lower and upper bounds of a and b, respectively. In general, the objective function is twice continuous differential, although it is not necessary for the global optimisation framework procedure, and with a proper local search algorithm also non-differentiable problems can be solved. On the other hand, one of the local search algorithms applies numerical derivatives calculated inside of it, so the user must not include subroutines for the calculation of derivatives. Therefore, the GLOBAL is a direct search method. GLOBAL has own termination criteria, so it stops when didn't find any new local minimum and all the sample points were clustered. Naturally it also stops when the number of find local minimums exceeds a given value. The derivative-free UNIRANDI local search method is part of GLOBAL package, while the BFGS (Broyden-Fletcher-Goldfarb-Shanno) local search procedure is part of the MATLAB optimisation package. GLOBAL has six parameters to set: the number of sample points, the number of best points selected, the stopping criterion parameter for local search, the maximum number of function evaluations for local search, the maximum number of local minima to explore, and the used local method. All these parameters have a default value.

5.3 Deterministic Global Optimisation- *DIRECT algorithm*

Both GA and GLOBAL algorithms are stochastic global optimisation methods and cannot guarantee the worst case is found, which is vital for ensuring the safety of unmanned vehicles. Therefore, the deterministic global optimisation method is investigated and applied to the moving OAS. DIRECT algorithm (DIviding RECTangles) is a deterministic global optimisation algorithms which is guaranteed to converge to the globally optimal if the objective function is continuous. DIRECT algorithm was developed by Jones et al [107] in 1993. The DIRECT algorithm was created in order to solve difficult global optimisation problems with bound constrained and a real-valued objective function. DIRECT method does not require any derivative information. It is a modification of standard Lipschitzian optimisation method. The DIRECT algorithm will globally converge to the minimal value of the objective function. This global convergence may come at the expense of a large and exhaustive search over the domain. This global search algorithm can be very useful when the objective function is a "black-box" function. DIRECT deals with problems on the form

$$\begin{aligned} & \min f(x) \\ s.t \quad & x_L \le x \le x_U \end{aligned} \tag{5.2}$$

where $f \in \mathbb{R}$ and $x, x_L, x_U \in \mathbb{R}^n$. DIRECT begins the optimisation by transforming the domain of the problem into a unit hyper-cube. That is,

$$\bar{\Omega} = \{x \epsilon R^N : 0 \leq x_i \leq 1\} \tag{5.3}$$

The functions then sampled at the center-point of this cube. Computing the function value at the center-point instead of doing it at the vertices is an advantage when dealing with problems in higher dimensions. The hypercube is then divided into smaller hyper-rectangles whose center-points are also sampled. Instead of using a Lipschitz constant when determining the rectangles to sample next, DIRECT identifies a set of potentially optimal rectangles in each iteration. All potentially optimal rectangles are further divided into smaller rectangles whose center-points are sampled. The procedure described above is performed for a predefined number of iterations. More details of the DIRECT algorithm can be found in [107, 108, 109].

5.4 Hybrid Optimisation

Based on the discussion in the previous section, DIRECT algorithm performs a global search of the variable space while identifying promising areas. In the last chapter, local optimisation results clearly show that the four benchmark studies considered for verification of OAS are non-convex problems and there is the chance to miss the true worst case because there is only little information available to choose a good initial starting point. If the initial guess is close to the true worst case, then the local optimisation methods can converge to the global optimum extremely quickly. In order to get the best solution from global and local optimisation algorithms, combining the two approaches of hybrid optimisation has been proposed [87]. The DIRECT algorithm including the local optimisation is referred to as H-DIRECT. The solution obtained from the DIRECT algorithm is considered as the initial starting point for the local optimisation method. The SQP is chosen for the local optimisation method, and *fmincon* function is applied to find the true worst case with initial starting point which is obtained from DIRECT algorithm. The hybrid optimisation attempts to find the best solution when the DIRECT convergence results are on the bounds of the uncertain parameter space.

5.5 Global Optimisation Worst-case Analysis Results

GA can be applied to the OAS to find the global minimum. The uncertain parameter set is considered here as the genetic representation, i.e. the chromosome. Each of the uncertainties corresponds to one gene. The selection function of roulette wheel is used for four case studies. The population size and crossover fraction are selected as default value of 20 and 0.8 respectively. The GLOBAL optimisation with UNIRANDI local search method is applied to find the global solution for OAS. The DIRECT algorithm is applied to the obstacle avoidance verification process. The DIRECT method requires no initial guesses but operates on the parameters upper and lower bounds. The DIRECT algorithm terminates as soon as it exceeds the given iterations. The reliability and efficiency of the global optimisation algorithms are compared and discussed.

5.5.1 Case Study-1: Simple Unicycle

The GA results with different starting points are given in Table.5.1. The number of generations is chosen as default value of 100, and the optimisation is terminated when the maximum number of generations exceeded. In order to compare the local and global optimisation results, same starting points are chosen (Table.4.1 and Table.5.1). At case-2, both *fmincon* and GA are converged to the nearly same optima. However, at case-1, *fmincon* converges to the worst-case parameters set of [4.4224, 0.06], while GA converges to the [5.734, 0.06]. Fig.5.1 shows the number of generations versus the best fitness and the mean fitness values at starting point at [5, 0.05]. The GLOBAL algorithm results with different sampling points are given in Table.5.2. GLOBAL algorithm is also converged to almost unique solution. DIRECT algorithm results at 100 iterations are given in Table.5.3. DIRECT algorithm number of iterations versus minimum distance to the obstacle is shown in Fig.5.2. All global optimisation algorithms are performed well for this case study as there are only two decision variables in this case study.

TABLE 5.1 GA RESULTS FOR A UNICYCLE OAS

GA	Starting point	m(kg)	J(kgm^2)	d_{min} (m)
Case-1	[4.5, 0.05]	5.734	0.06	0.9378
Case-2	[5.5, 0.048]	5.192	0.06	0.9382

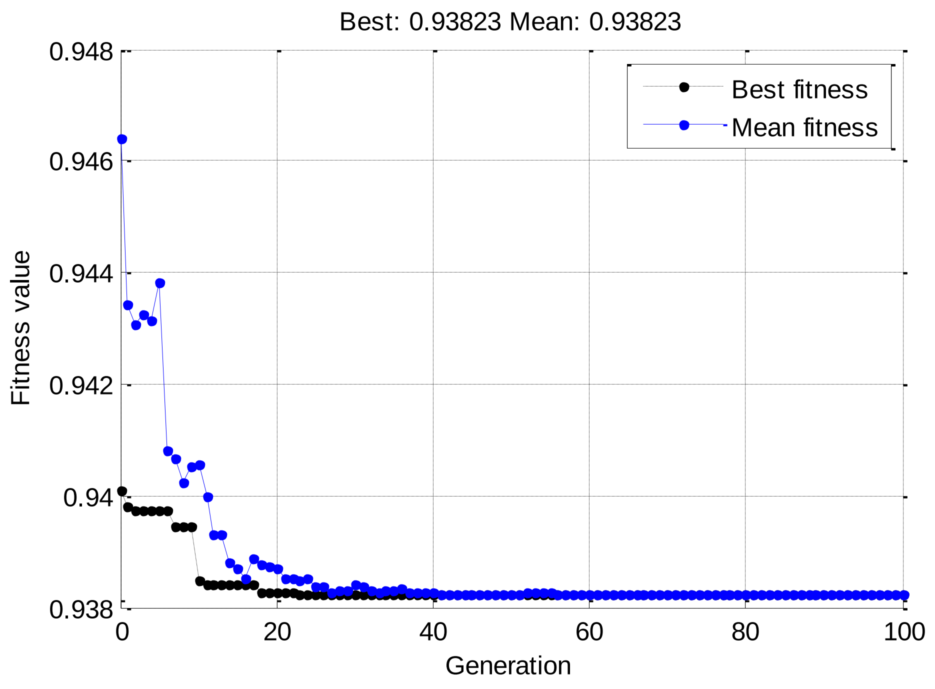


Figure 5.1: GA- No of generations vs. fitness value

TABLE.5.2 GLOBAL RESULTS FOR UNICYCLE OAS

Algorithm	No of SAMPLE	m(kg)	J (kgm^2)	d_{min}(m)
GLOBAL-with *UNIRANDI*	20	5.9734	0.06	0.9377
GLOBAL-with *UNIRANDI*	50	5.9734	0.06	0.9378

TABLE.5.3 DIRECT RESULTS FOR UNICYCLE OAS

Algorithm	Iterations	m(kg)	J(kgm^2)	d_{min} (m)
DIRECT	100	5.939	0.06	0.9378

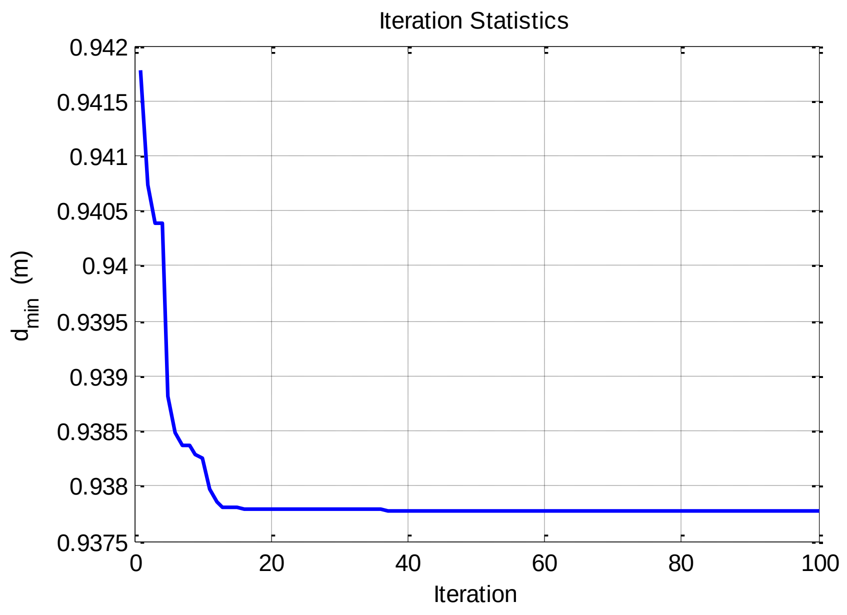


Figure 5.2; DIRECT-No of iterations vs. function value of d_{min}

Final step of the verification process of OAS is to validate the proposed algorithm results. Therefore, these worst-case condition and worst-case parameters are further validated using with simulation response which is shown in Fig.5.3. The worst-case minimum distance to the obstacle d_{min} is *0.9378m* which is greater than the safety radius. This response shows that the obstacle avoidance algorithm and the controller are working correctly for worst-case parameters. At worst-case parameters, the simulation response with two obstacles is shown in Fig.5.4. Therefore, the proposed controller for one obstacle is functioning correctly for two obstacles at worst-case parameters.

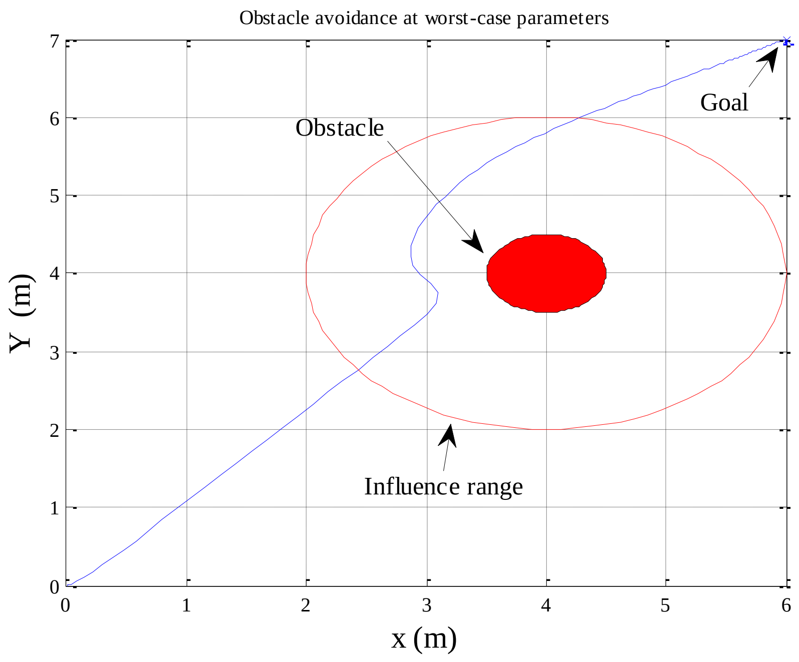


Figure 5.3: Simulation result at worst-case parameters

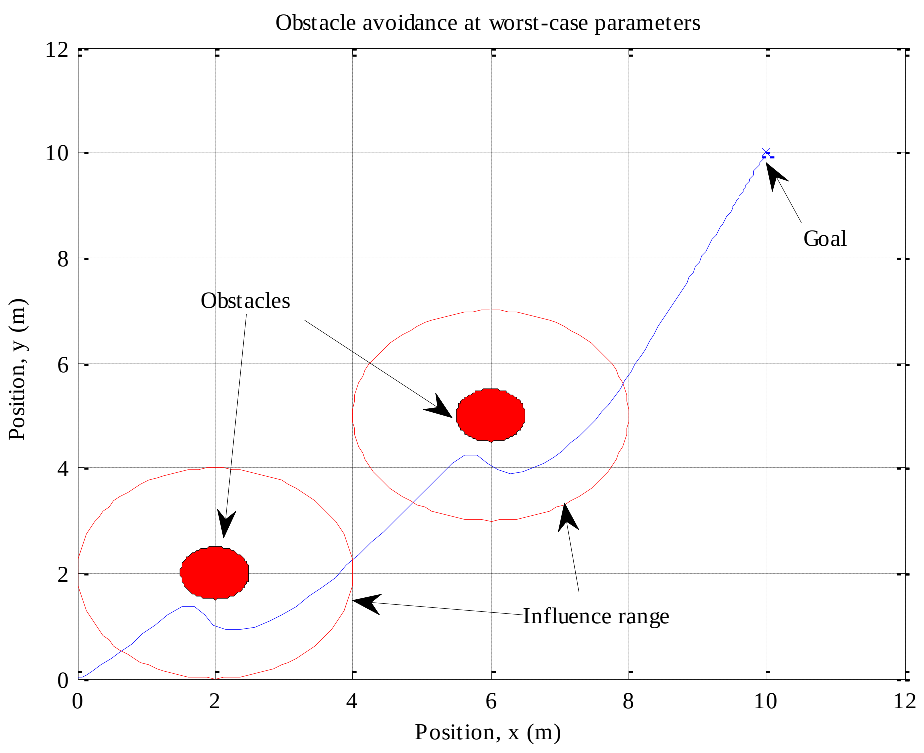


Figure 5.4: Simulation response at worst-case parameters

5.5.2 Case Study-2: Pioneer 3-DX

Stochastic algorithm including GA and GLOBAL and deterministic algorithm such as DIRECT are applied to the moving OAS to find the worst-case condition and worst-case parameters set. Eight design variables are restricted within a lower and an upper bound during this process. The GA optimisation is terminated after given iterations (100). For GLOBAL optimisation with UNIRANDI local search method, the sampling points are chosen as 200. The DIRECT algorithm terminates as soon as it exceeds the given iterations of 200.

A comparison of the minimum distance to obstacle before and after the optimisation is given in Table.5.4. A significant change in minimum distance to obstacle is seen after the optimisation. All optimisation algorithms are performed in MATLAB 2011b and Intel (R) Core(TM) 2 Duo CPU (3.16GHz). The minimum distance to the obstacle with the DIRECT, GA and GLOBAL algorithms are very closer. GLOBAL took 1112 functions evaluation with 200 sampling points while DIRECT took 8751 function evaluations. GA took 2 hours and 26 minutes to converge to the global minimum while GLOBAL and DIRECT algorithms took around 5 hours and 20 minutes. GA performs faster than other two algorithms; however, DIRECT algorithm can guarantee the global minimum.

TABLE.5.4. COMPARISON OF WORST-CASE CONDITION, d_{min}(m)

Before optimisation d_{min} (m)	After Optimisation d_{min} (m)		
Norminal Case	**DIRECT**	**GLOBAL**	**GA**
7.6668	5.8726	5.8719	5.8758

Final values of eight design variables after optimisation are shown in Table.5.5. It can be seen that the mass is greatly increased from 18 to 34 *kg*. And also, there are huge differences in other parameters. All three global algorithms are converged to nearly same values. The history of iteration versus fitness value for the DIRECT algorithm is shown in Fig.5.5. This figure shows that the fitness value of d_{min} is almost the same from iteration 50 to 200.

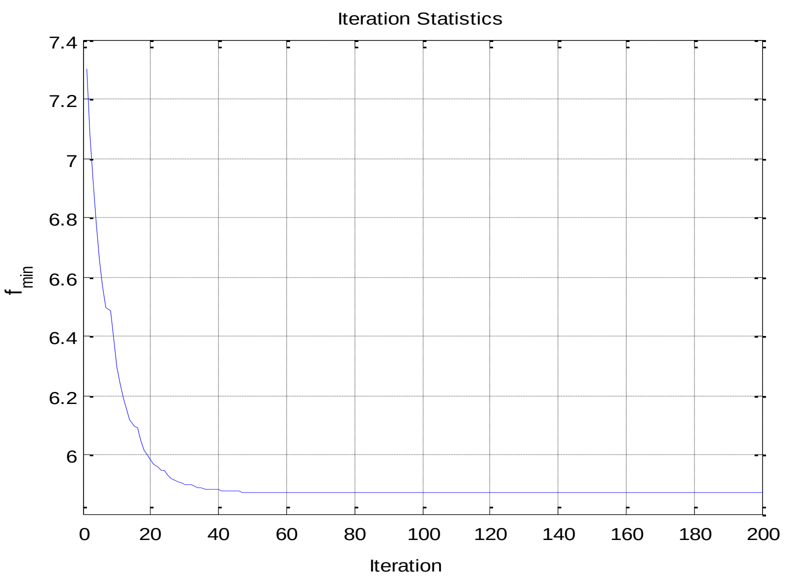


Figure 5.5: *DIRECT* algorithm- Iteration vs. Fitness value

TABLE.5. 5. COMPARISON OF WORST-CASE PARAMETERS VALUES

Design Variable	Initial Value	Final Value		
		DIRECT	**GLOBAL**	**GA**
m	18	33.994	34	33.989
B_e	0.8	0.48	0.48	0.4806
δ_u	0.2	0.8999	0. 90	0.8997
δ_ω	0.5	0.1	0.1	0.1
I_e	2	3.7975	3.7998	3.7978
$\bar{u}^s$	0.05	0.02	0.02	0.02
Δ_x	0	0.4999	0. 5	0.4993
Δ_y	0	-0.5	- 0. 5	- 0.5

In order to get the best results, H-DIRECT algorithm is also applied to the moving OAS. The global solution found for this case study is [m, B_e , δ_u , δ_ω , I_e , $\bar{u}^s$, Δx , Δy] = [34.0, 0.48, 0.90, 0.1, 3.80, 0.02, 0.50, -0.5]. Minimum distance to the obstacle obtained from H-DIRECT is 5.8722m, and it converges to nearly same solution of DIRECT algorithm.

Based on optimisation-based verification method, the optimized minimum distance to the obstacle d_{min} is decreased from 7.6668 to 5.8726m. The performance of the moving obstacle avoidance algorithm at worst-case parameters is checked with simulation response which is shown in Fig.5.6. The worst-case minimum distance to the obstacle d_{min} is 5.8726m which is greater than the specified safety radius of the obstacle. This concludes that the moving obstacle avoidance algorithm and the controller provide adequate performance at the worst-case parameters. Furthermore, in the presence of all the described variations and uncertainties, the safety margin for anti-collision is respected. The time versus distance to the obstacle at the nominal and worst-case parameters is shown in Fig.5.7. It clearly shows that there is a significant difference in the minimum distance to the obstacle at nominal and worst-case parameters during the manoeuvre.

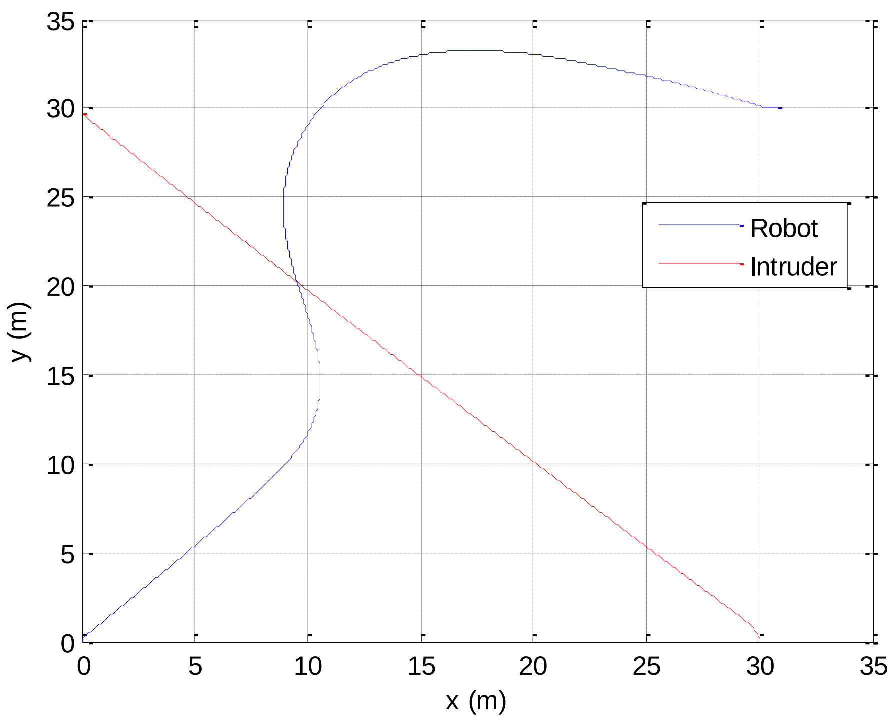


Figure 5.6: Simulation results at worst-case parameters, t = 40 sec

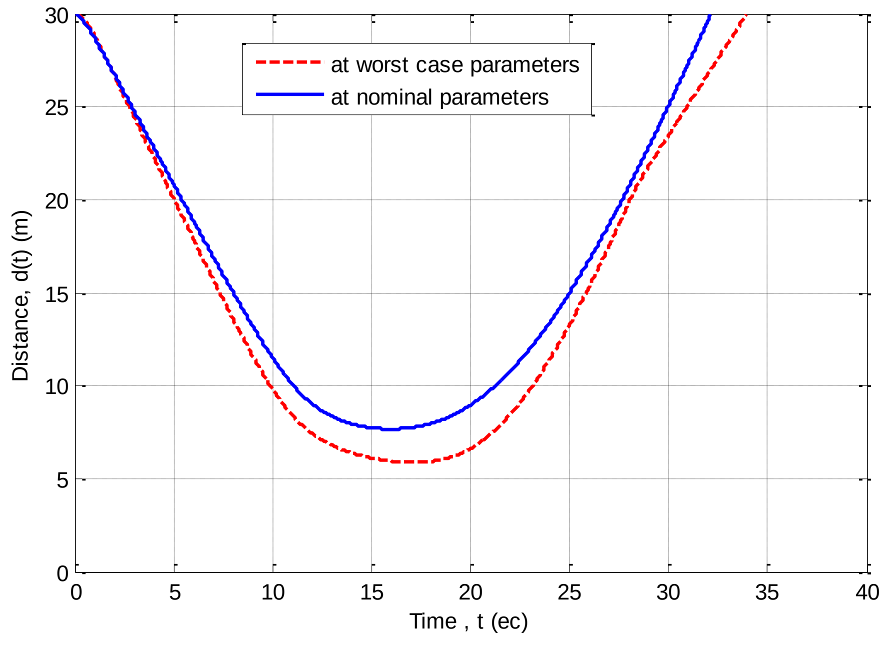


Figure 5.7: Time vs distance to the obstacle at nominal andworst-case parameters

5.5.3 Case Study-3 : UAV model-Static OAS

GA can be applied to the UAV collision avoidance system to find the global minimum. The optimization is terminated after 51 iterations because the convergence accuracy is met. The

GA results with different starting points are given in Table.5.6. GA only took 12 sec to converge to the global solution with different starting points. Fig.5.8 shows the number of generations versus the best fitness and the mean fitness values at starting points [1.52, -1.13, 11.82]. The GA optimisation is terminated after 51 iterations because the best solution achieved is less than the defined accuracy level (TolFun and TolCon) at 10^{-6}. The GLOBAL optimisation with UNIRANDI local search method is applied to find the global solution for the OAS. The results with different numbers of the sampling points are given in Table.5.7. GLOBAL algorithm gives the nearly same solutions with different number of sampling points. It takes 1112 functions evaluation with 200 sampling points while 9810 functions evaluations with 500 sampling points. 36 local minimum are found at 500 sampling points while 8 found at 200 sampling points.

TABLE.5.6. GA RESULTS FOR A UAV OBSTACLE AVOIDANCE SYSTEM

Algorithm	Starting point [m, $C_{m\delta e}$, $C_{\delta t}$]	*m(kg)*	$C_{m\delta e}$	$C_{\delta t}$	$d_{min}(m)$	Time
GA	[1.52, -1.13, 11.82]	2.2773	-1.0283	10.9734	27.5018	12 sec
GA	[1.71, -1.13, 13.409]	2.28	-1.0177	10.9905	27.5032	12 sec

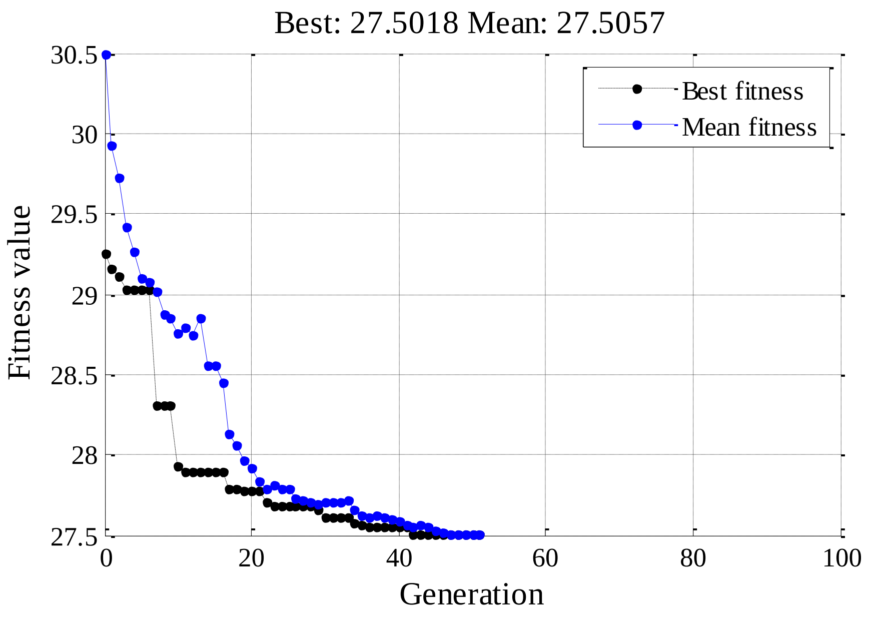


Figure 5.8: No of generations vs. Fitness value

TABLE.5.7. GLOBAL RESULTS FOR UAV OBSTACLE AVOIDANCE SYSTEM

No of SAMPLE	*m (kg)*	$C_{m\delta e}$	$C_{\delta t}$	$d_{min}(m)$	Fun.Evalu taken	Time
200	2.2063	-1.0265	11.2221	27.748	1112	13 mins
500	2.1798	-1.0171	10.971	27.494	9810	2 hours 15 mins

The DIRECT algorithm is applied to the UAV obstacle avoidance verification process, and the results are given in Table.5.8. The DIRECT method requires no initial guesses but operates on the parameters upper and lower bounds. The DIRECT-history of iteration versus fitness value is shown in Fig.5.9. All optimisation algorithms are performed in MATLAB 2010a and Intel (R) Core(TM) 2 Duo CPU (3.16GHz). DIRECT takes 3 hours 35 minutes to converge to the global minimum. Compared to the stochastic global algorithms, GA and GLOBAL algorithm are performed well for this case study, but GA performs faster. GA terminates the search process However, these are stochastic global algorithms and there is no confidence to establish the true worst case. The DIRECT algorithm can guarantee finding the worst case, but the computation time is high. The worst-case parameters found by H-DIRECT is [m, $C_{m\delta e}$,$C_{\delta t}$] = [2.28, -1.017, 10.976], and worst-case distance to the obstacle is 27.49m. H-DIRECT and DIRECT are converged to the same solution. There is no improvement in the H-DIRECT search for this case study.

These worst-case condition and worst-case parameters identified in the verification process are further validated with simulation response shown in Fig.5.10. The time versus distance to the obstacle at the nominal and worst-case parameters is shown in Fig.5.11. The worst-case minimum distance to the obstacle d_{min} is 27.4982*m* which is greater than the specified safety radius of the obstacle. This concludes that the obstacle avoidance algorithm and the controller provide adequate performance at the worst-case parameters.

TABLE.5.8. DIRECT RESULTS FOR A UAV OBSTACLE AVOIDANCE SYSTEM

Algorithm	Iteration	*m (kg)*	$C_{m\delta e}$	$C_{\delta t}$	d_{min} *(m)*	Fun.Evalu taken	Time
DIRECT	500	2.28	-1.017	10.976	27.498	18505	3 h 35m

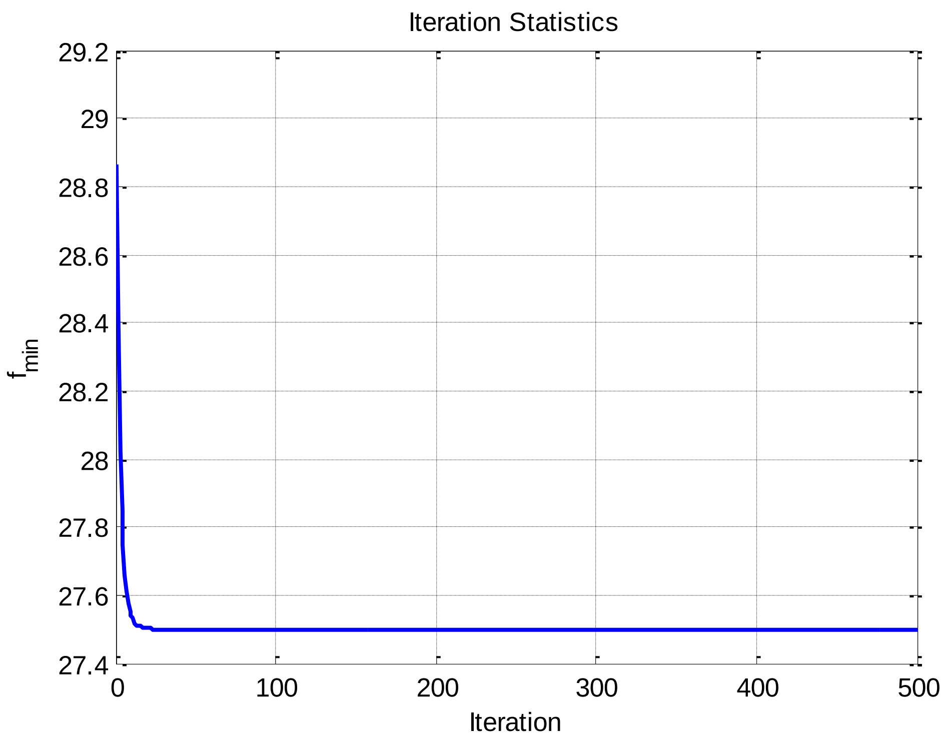


Figure 5.9: ***DIRECT* algorithm- Iteration vs. Fitness value**

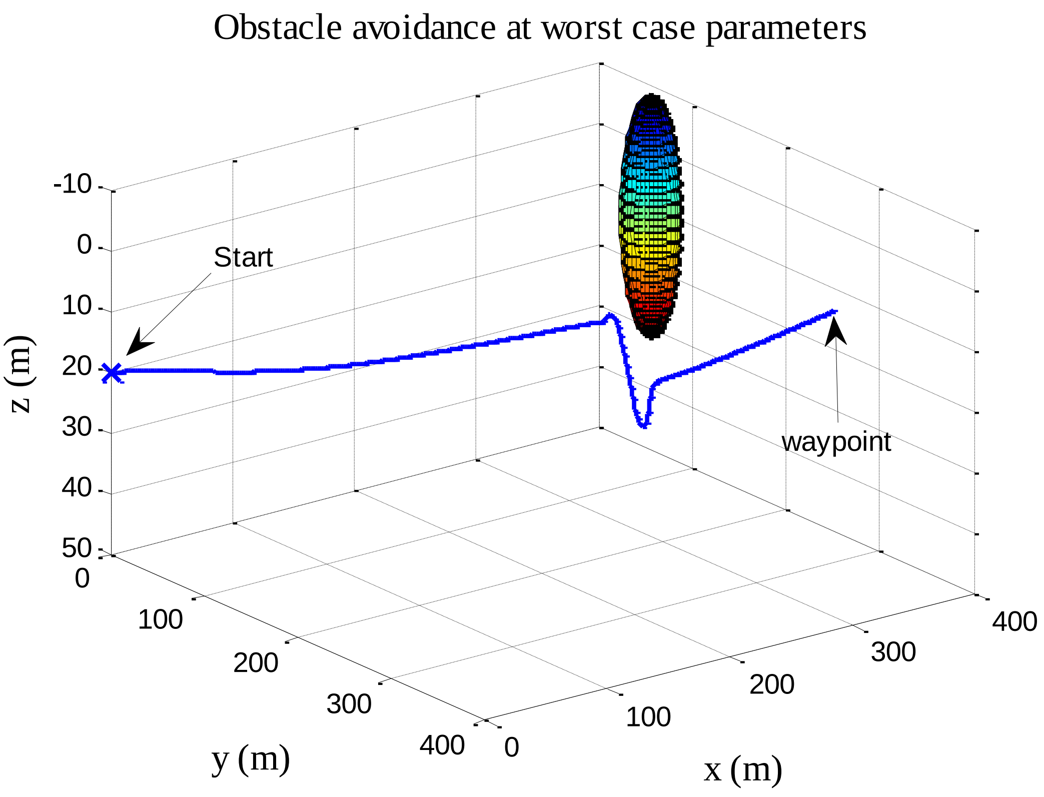


Figure 5.10: Simulation response at worst-case parameters

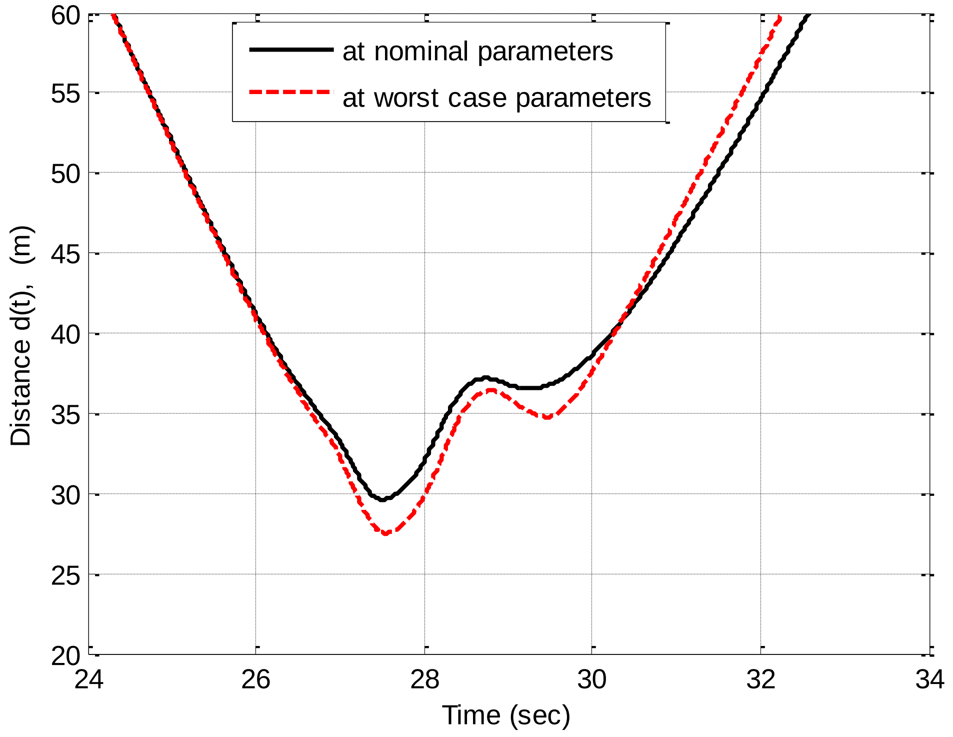


Figure 5.11: Time vs distance to the obstacle at nominal and worst case parameters

5.5.4 Case Study-4 : UAV model- Moving OAS

The optimisation-based verification process of OAS is applied to verify the collision avoidance algorithms for UAVs. To demonstrate the concept, a 6DOF UAV model is used in the case study with a designed collision avoidance algorithm, and mass and two aerodynamic coefficients variations are considered for the verification purpose which were described in chapter 3.

It shall be highlighted that in developing collision avoidance algorithms using the potential field method, the UAV is considered as a mass point and only the kinematic model is used. However in real implementation of collision avoidance maneuver, the UAV has to be controlled to follow the desirable total velocity and attitude. Therefore, the UAV dynamics and the influence of the inner loop controllers for tracking reference speed and attitude provided by the collision avoidance algorithm must be taken into account in order to fully understand the behavior of the collision avoidance algorithm. This is particularly important for very close maneuvers like collision avoidance. The work presented in the case-studies provide a framework of taking into account the different levels of the model complexity used in the different stages of autonomous control development. It can significantly improve the efficiency of the verification process by automatically searching the worst cases without the need to exhaustively evaluate all possible combinations of variations.

A comparison of the minimum distance to obstacle before and after the optimisation is given in Table.5.9. A significant change in minimum distance to obstacle is seen after the optimisation. Final values of three design variables after optimisation are shown in Table.5.10. All three global algorithms are converged to nearly same values. All optimisation algorithms are performed in MATLAB 2012b and Intel (R) Core(TM) 2 Duo CPU (3.16GHz). All three algorithms are converged to the same solution for this case study. GLOBAL took 2519 functions evaluation with 500 sampling points while DIRECT took 6061 function evaluations at 200 iterations. GA took 1 hours and 27 minutes to converge to the global minimum while DIRECT algorithms took around 5 hours and 18 minutes. GLOBAL took around 2 hours 32 minutes. GA performs faster than other two algorithms; however, DIRECT algorithm can guarantee the global minimum. The history of iteration versus d_{min} values is shown in Fig.5.12. This figure shows that the fitness value of d_{min} is almost same from iteration 40 to 200. Fig.5.13 shows the function evaluations versus d_{min} values.

TABLE.5.9. COMPARISON OF WORST-CASE CONDITION,d_{min}(m)

Before optimisation d_{min} (m)	After Optimisation d_{min} (m)		
Norminal Case	**DIRECT**	**GLOBAL**	**GA**
59.23	57.90	57.90	57.90

TABLE.5. 10. COMPARISON OF WORST-CASE PARAMETERS VALUES

Design Variable	Initial Value	Final Value		
		DIRECT	**GLOBAL**	**GA**
m	1.9	1.52	1.52	1.52
$C_{m\delta e}$	-1.13	-1.356	-1.356	-1.356
$C_{\delta t}$	12.19	9.752	9.752	9.752

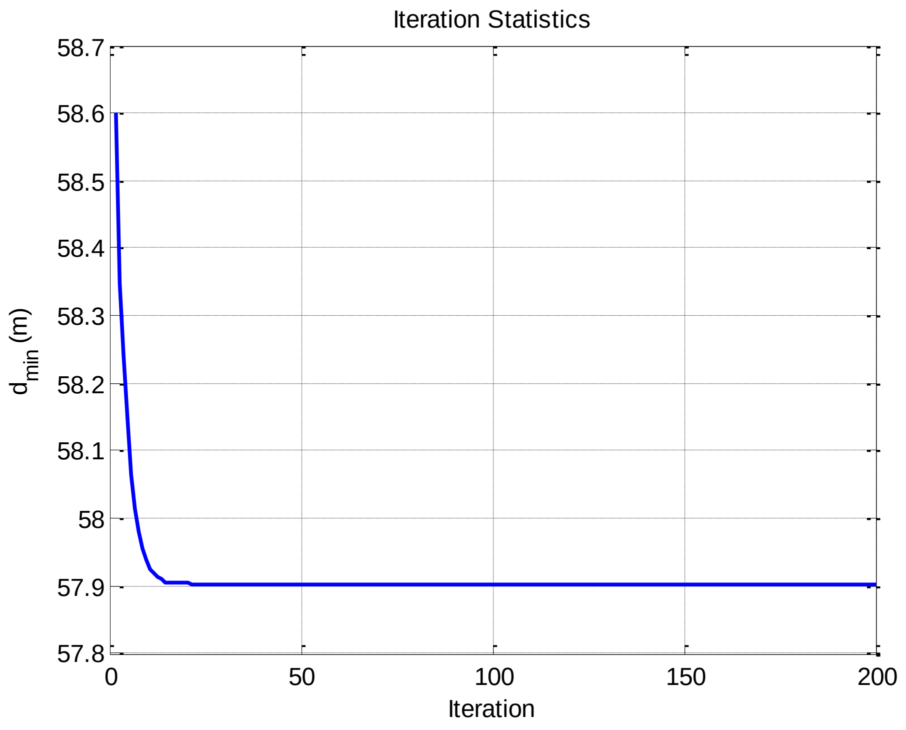


Figure 5.12: *DIRECT* algorithm- Iteration vs. d_{min}value

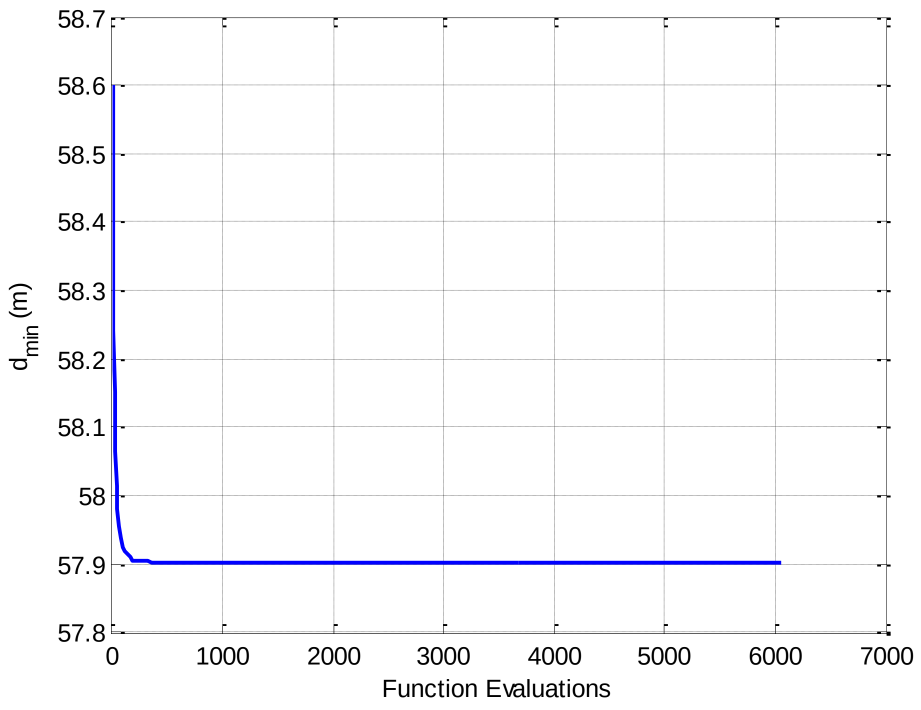


Figure 5.13: *DIRECT* - Function evaluations vs. d_{min} value

The worst-case parameters found by H-DIRECT is [m, $C_{m\delta e}$,$C_{\delta t}$] = [1.52, -1.356, 9.752], and worst-case distance to the obstacle is 57.9m. H-DIRECT and DIRECT are converged to the same solution.

The performance of the moving obstacle avoidance algorithm at worst-case parameters is checked with simulation response which is shown in Fig.5.14. Based on optimisation-based verification method, the optimized minimum distance to the obstacle d_{min} is decreased from 59.23 to 57.90m which is greater than the specified safety radius of the obstacle. This concludes that the moving obstacle avoidance algorithm and the controller provide adequate performance at the worst-case parameters. Furthermore, in the presence of all the described variations and uncertainties, the safety margin for anti-collision is respected. The time versus distance to the obstacle at the nominal and worst-case parameters is shown in Fig.5.15. It clearly shows that there is a significant difference in the minimum distance to the obstacle at nominal and worst-case parameters during the manoeuvre.

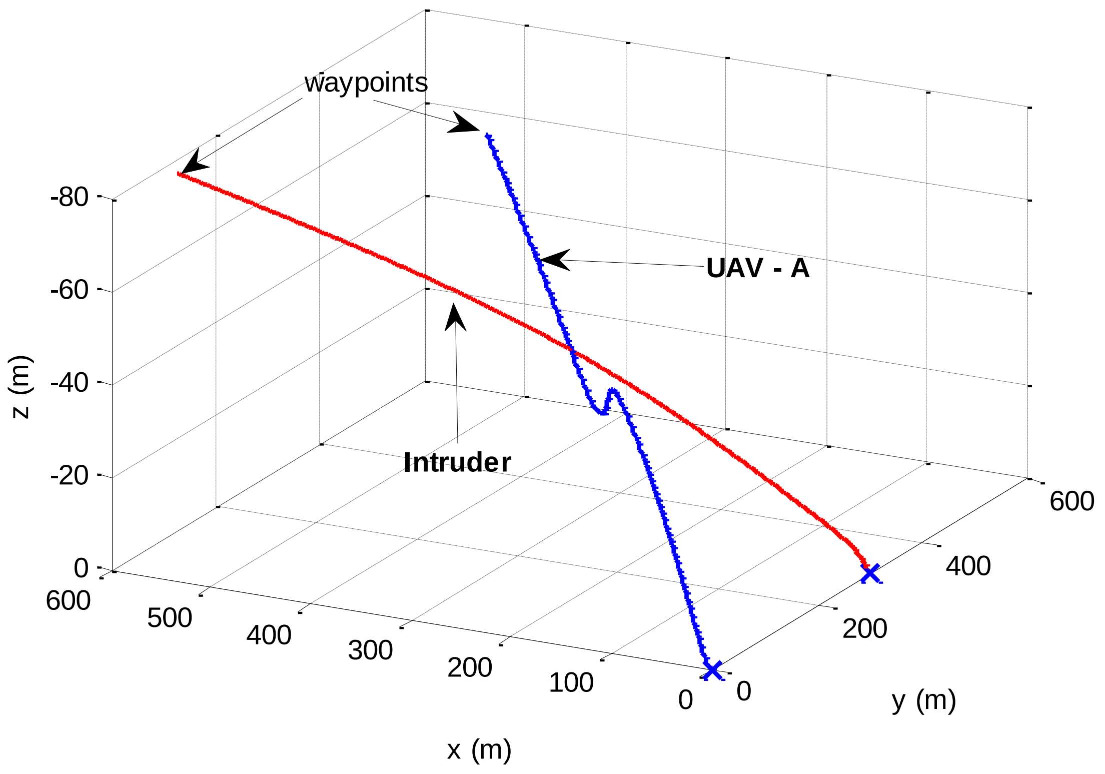


Figure 5.14: Simulation response at worst-case parameters

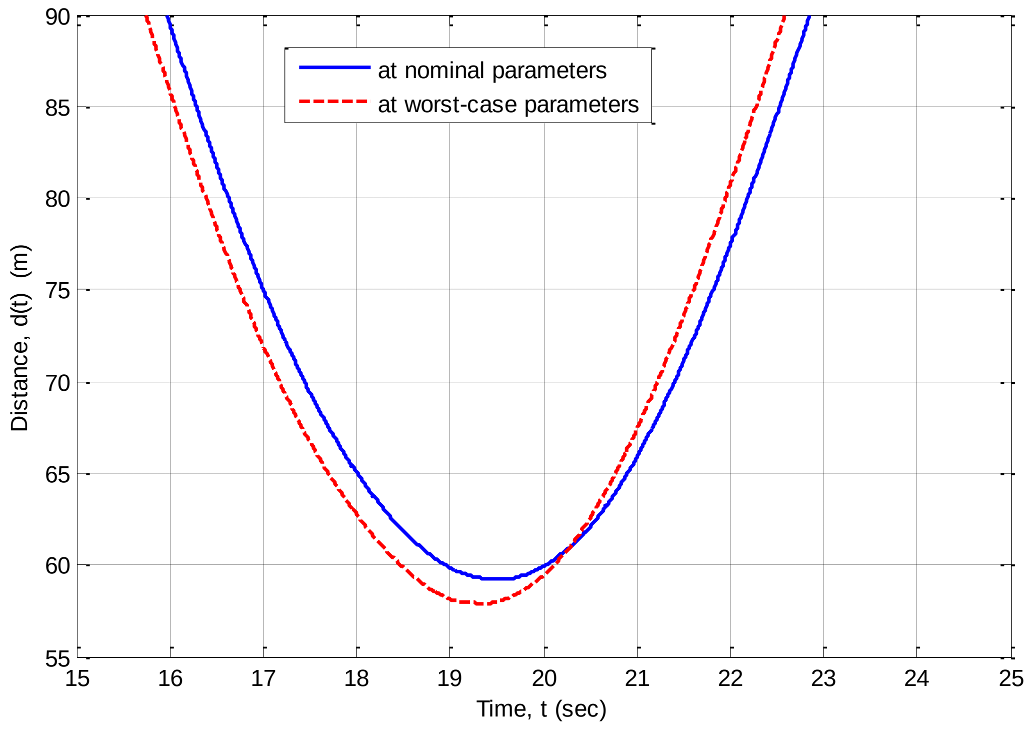


Figure 5.15: Time vs distance to the obstacle at nominal and worst case parameters

5.6 Conclusion

For unmanned vehicles verification, the worst-case analysis is highly important and industry needs for new methods to verify the OAS in the presence of all possible parameters variations and failure conditions. In this chapter, the safety analysis of collision avoidance systems is presented. The key idea in this verification approach is that it is not necessary for an optimisation algorithm to evaluate a cost function over all possible solutions in order to find the optimal solution. However different from many optimisation problems, it is important to find all the possible worst cases in order to verify safety critical functionalities like obstacle avoidance. This requires an optimisation algorithm that converges to the global optimal solution.The optimisation-based verification process method has applied for verification of collision avoidance algorithms for unmanned vehicles. The optimisation-based approach is developed to find the worst cases which are defined by the minimum distance to the obstacle in the presence of all possible described variations. In the last chapter, different worst cases were identified when the local optimisation starts from different initial conditions. Therefore, the local optimisation is not suitable for verification of collision avoidance algorithms for these case studies. To overcome this problem, global optimisation algorithms are required and discussed in this chapter.

Stochastic global optimisation algorithms including GA and GLOBAL methods were applied to the problem. In order to understand the proposed method of optimisation-based verification algorithm, very simple unicycle mobile robot was considered. Static obstacle avoidance algorithm with artificial potential field method is verified within the parameters range. Only two uncertain parameters of mass and inertia were defined within the lower and upper bounds. Both algorithms are performed well for this case study. Then moving collision avoidance algorithm using with potential field method is applied to the more complicated Pioneer 3DX robots. Initial robustness analysis were carried out, and most significant eight uncertain parameters including obstacle sensor data uncertainties were chosen within the lower and upper bounds. It is a very challenging task to find the true worst case scenario for this case study because this is the verifying moving OAS with more design variables including sensor data uncertainty. Furthermore, it is a non-linear analysis problem in the search space with many local minima. It seems to be a real-world problem. The GA and GLOBAL algorithms are almost converged to the same global solution. However, GA is very faster than GLOBAL algorithm. GA took only 2 hours 20 minutes to converge while GLOBAL took around 5 hours 20 minutes. After considering the verification of OAS application for UGVs, this work was extended to the UAVs. Based on a 6 Degree of Freedom (6DoF) kinematic and dynamic model of a UAV, the path planning and collision avoidance algorithms were developed in 3D space. Static and moving obstacle avoidance algorithms were developed for UAVs using with potential methods. Proposed OAS was verified at nominal parameters, and then clearance criterion of minimum distance to the obstacle was defined as the objective function in the time domain. Mass and two aerodynamic coefficients variations were considered for the verification purpose. The convergence results of GA and GLOBAL for these cast studies are almost closer. And again, GA performs faster the GLOBAL algorithm.

These stochastic algorithms results are very promising and of significant industrial interest, however, multiple trials are required to provide confidence in results. To address this problem, a deterministic global optimisation algorithm, DIRECT algorithm was applied to the problem of non-convex OAS verification. The DIRECT algorithm achieved almost the same quality of solution as that obtained from GA and GLOBAL. Compared with these stochastic global optimisation algorithms in this study, the DIRECT algorithm can guarantee finding the worst case, although it takes more time to converge. In order to increase the efficiency of the results, the hybrid optimisation was considered. In the interest of convergence proof of true worst-case results, the H-DIRECT method was only considered in this chapter. The results are shown that DIRECT and H-DIRECT algorithms were converged to nearly same solutions.

Potential field method was chosen for this verification study as it is a simple and widely used in the industry. These collision avoidance algorithms are verified using with proposed optimisation-based verification approach in this thesis. The results were shown that there is a significant change in minimum distance to obstacle is seen after the optimisation. Figures 5.7, 5.11, and 5.15 clearly demonstrate the interest of this purpose of study. The worst-case minimum distance to the obstacle obtained from this approach which is greater than the specified safety radius of the obstacle. This concludes that the obstacle avoidance algorithm and the controller provide adequate performance at the worst-case parameters. Therefore, exciting static and moving potential field methods function correctly with proposed controller at nominal and worst-case parameters. In order to further validate the proposed optimisation-based verification of OAS algorithm, the Monte Carlo analysis will be carried out in the next chapter as Monte Carlo method is a widely used in the industrial practice to identify the worst cases.

Chapter 6

Comparison with Monte Carlo Analysis

6.1 Introduction

Monte Carlo Method (MCM) is widely used method and industrial practice to identify the worst cases. Monte Carlo simulation involves checking various criteria in the presence of random values of uncertain parameters within the lower and upper bounds. Monte Carlo simulation is carried out to provide a benchmark comparison of the proposed automatic worst case search methods in this chapter. A known weakness of this approach is that there is no absolute guarantee to find the global worst case. MCM is a probabilistic method and it requires computer calculations for generating pseudo-random numbers and for evaluating the model a large number of times. MCM performs a characterization of the quantities measured based on the random sampling of the probability distribution functions (PDFs). MCM is used to evaluate the minimum distance to the obstacle within the described parameter variations. The PDFs for each input quantities are assigned for four case studies, and a number of Monte Carlo trials are carried out in this chapter.

The Monte Carlo technique is a numerical approach, providing a numerical approximation to the distribution function of the output quantity. The implementation of MCM is in MATLAB environment. There are three stages of uncertainty evaluation in the MCM procedure [110 ; 115]: formulation, propagation and summarizing. In the formulation stage, the output quantity Y is defined and the input quantities $X=[X_1,.....X_N]^T$ are determined. Then a model relating Y and X is developed, and the PDFs are assigned to X_i. In the propagation stage, the PDFs for X_i are propagated through the model to obtain the PDF for Y. In summarizing stage, the expectation of Y is obtained from the PDF for Y.

Monte Carlo simulation is carried out to demonstrate that the proposed automatic search methods provide a significant advantage over random sampling approaches.

6.2 Monte Carlo Simulation Results

MCM is applied to the OAS of simple unicycle mobile robot which is presented in section-3.2. There are two input quantities considered in this case study: mass and inertia. Each uncertain parameter is allowed to vary within ±*20%* of its nominal value. There is no probability information is available within the interval ranging. Thus, a uniform PDF can be associated with these input parameters within the minimum and maximum limits. Monte Carlo simulation is executed with 1,000 runs to find the worst case and the results are shown in Fig.6.1. The minimum distance to the obstacle d_{min} obtained by MCM is 0.9398m which is same as obtained by proposed optimisation-based verification approach. MCM performs well for this case study.

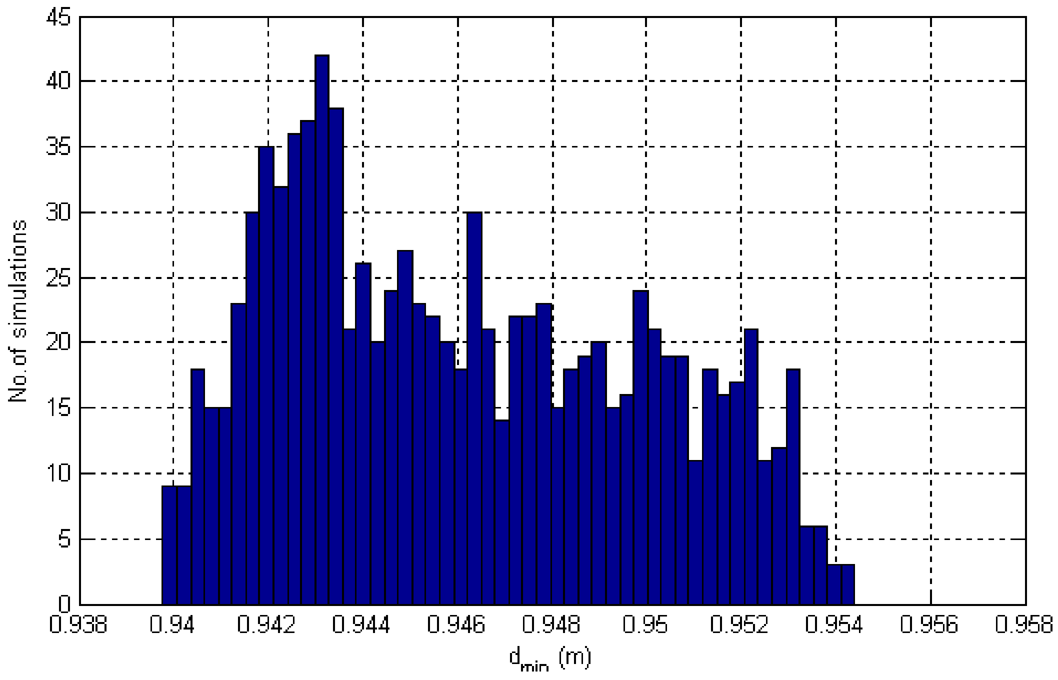


Figure 6.1: Monte Carlo simulations results for unicycle OAS

Next Monte Carlo simulation is executed with 5000 runs to find the worst case scenario for the OAS of Pioneer 3DX robot (section.3.3). A rectangular uniform distribution is assigned to the eight uncertain parameters within their corresponding lower and upper bounds. The minimum distance to the obstacle d_{min} at the worst case obtained by MCM is 6.43m (See. Fig.6.2) while that identified by the optimisation based automatic search method is 5.8m.The worst case condition obtained from the MCM is not the true worst case and there is a high chance of missing the true worst case solution in this approach. Therefore, the proposed automatic worst case analysis approach provided a more efficient and reliable verification method for the collision avoidance systems.

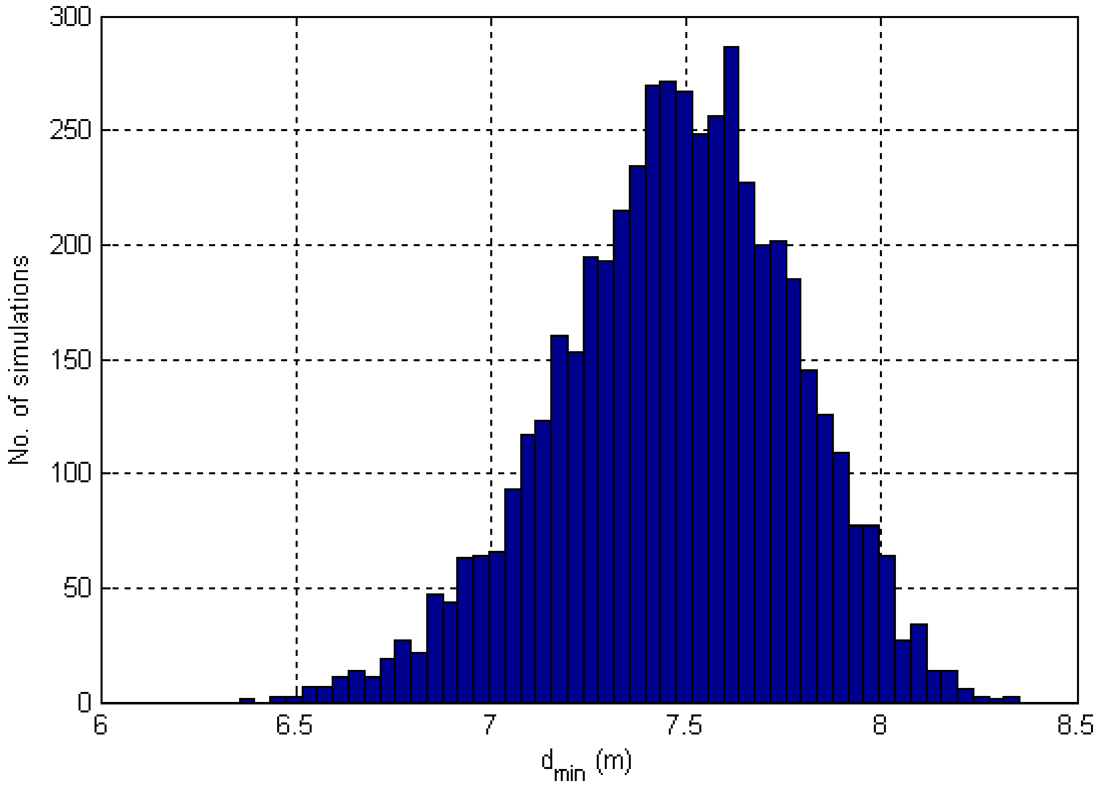


Figure 6.2: Monte Carlo simulations results for Pioneer 3DX OAS

Then Monte Carlo analysis is performed for comparison of criterion d_{min} for UAVs OAS. Static and moving OAS for UAVs are reconsidered (section.3.4 & 3.5), and all worst case search parameters are assumed to be uniformly distributed over the defined lower and upper bounds. Monte Carlo simulation is executed with 5000 sample size for both case studies. Figs. 6.3 - 6.4 show the criterion values and corresponding cumulative distribution function for the static and moving OAS, respectively. For the static OAS, minimum distance to the obstacle d_{min} is obtained as 27.58 by Monte Carlo analysis, and for the moving OAS, d_{min} is obtained as 57.99m, which are same as obtained from optimisation search method. Therefore, MCM are performed well for these two case studies.

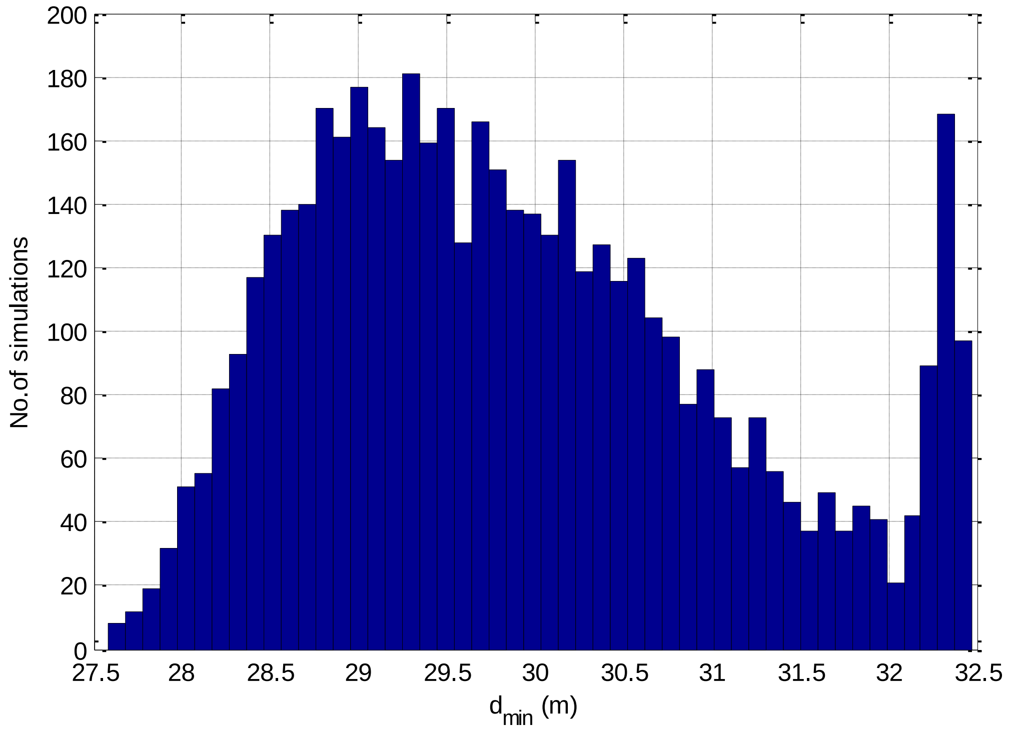


Figure 6.3: Monte Carlo simulations results for UAV static OAS

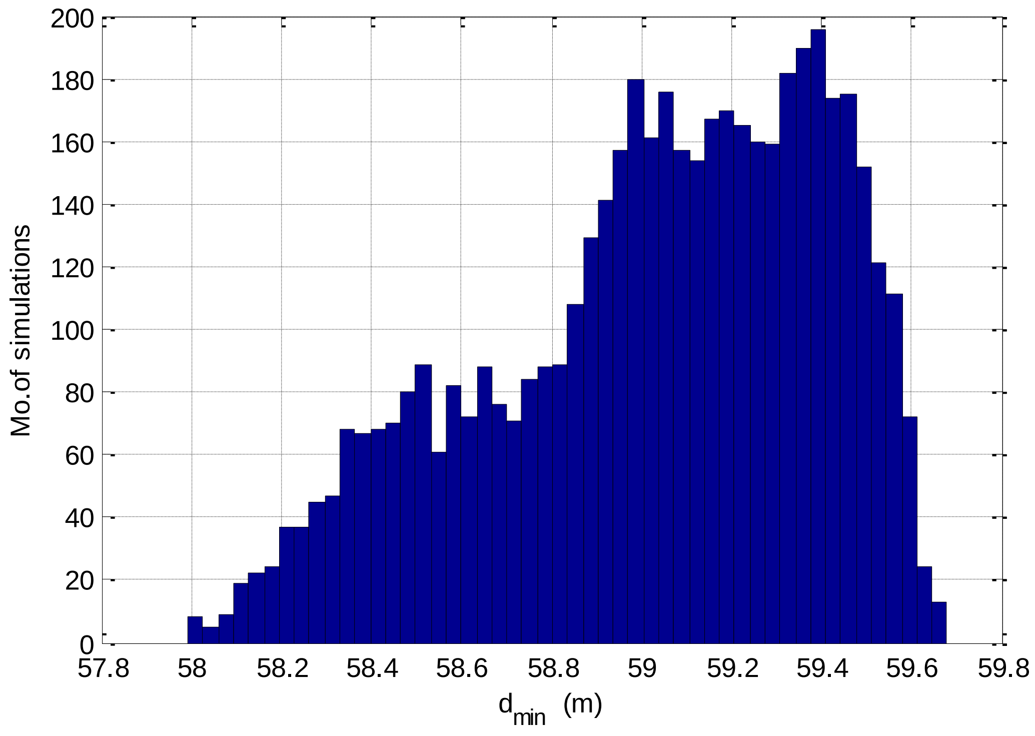


Figure 6.4: Monte Carlo simulations results for UAV moving OAS

6.3 Conclusion

In this chapter, the Monte Carlo simulations were carried out to compare the proposed optimisation verification method. MCM was applied to the four case studies. A rectangular uniform distribution is assigned to uncertain parameters within their corresponding lower and upper bounds. For case studies 3.1, 3.3 and 3.4, minimum distance to the obstacle is obtained by Monte Carlo analysis which is same as obtained from optimisation search method. MCM performs well for these three case studies. However, for case study 3.2, the minimum distance to the obstacle at the worst case obtained by MCM is not same as from optimisation search method. Therefore, the results clearly demonstrate that the optimisation-based worst-case analysis methods achieve better performance than the Monte Carlo approach for this case study.

Chapter 7

Verification of Decision-making Algorithm for Autonomous Collision Avoidance

7.1 Introduction

The robustness analysis is to determine under what conditions the collision avoidance system satisfies the safety performance for all admissible uncertainty. Since many problems in robustness analysis and synthesis can be formulated as the minimisation/maximisation of an objective function with respect to the uncertain parameters, optimisation-based robustness analysis method will result in powerful tools that can address real engineering control problems. In the last chapters, the potential field method for static and moving obstacle avoidance algorithms were chosen for the verification studies and worst-case analysis results were presented. In this chapter, the different method of collision avoidance algorithm which is called as Decision-making collision avoidance algorithm is chosen for the verification study and investigated the worst-case scenarios. In order to verify the decision-making collision avoidance algorithm in the presence of all possible uncertainties, the problem of avoidance of conflict or collision between two aircrafts in a 3-D environment utilizing current positions and velocities using a geometric approach is studied in this chapter. An optimisation-based worst-case analysis method is applied to the decision-making collision avoidance algorithm for pair-wise non-cooperative aircraft.

Aircraft mid-air collision is still a major problem with the increasing emerging traffic of small business aircraft. In order to increase aircraft capacity in the airspace, a robust autonomous collision avoidance system must be designed. Generally, aircraft must maintain a minimum airspeed to guarantee a sufficient lift to remain aloft. Air speed restriction and turning rate restrictions on an aircraft limit the manoeuvring zone. Therefore, the problem of designing an autonomous collision avoidance algorithm is more critical when applied to aircraft. Collision avoidance systems require both obstacle detection sensors and a collision avoidance algorithm that utilises the information obtained from the sensors to determine a path through the obstacle field. The aircraft Conflict Detection and Resolution (CD&R) has been the object of intensive research over the past several years. Autonomous Collision Avoidance (ACA) system is composed of on-board detection sensors and decision-making algorithms. Based on a geometric approach, an analytical solution to a proper kinematic optimisation problem is derived in [111]. This solution implies the simultaneous change of all control variables: speed module, track and slope angles. This approach does not require the solution of any programming problem, thus resulting suitable for real-time applications.

A comprehensive survey of conflict detection and resolution approaches is presented in [112]. Most of the methods presented in this literature are not suitable for real-time applications, because non-deterministic computational time needed for taking a decision. In [113], an analytical geometric approach to the problem of collision between two aircrafts is presented for a planar scenario. The problem of collision between two aircraft using with mixed geometric and collision cone approach is discussed in a 3D environment [114]. In [79], three different conflict resolution strategies, each one involving a single control variable – χ (lateral-directional control), γ (longitudinal control) or V (speed control), were investigated. In this approach, Conflict resolution involves only one control variable at a time. In [111], on the other hand, a real 3D analytical conflict resolution solution is designed. This solution implies the simultaneous change of all control variables. This analytical solution opens the way to the application of assessed non-linear control analysis and synthesis techniques for a-priori performance and stability robustness evaluations [111, 116]. Therefore, this collision avoidance algorithm will be studied and verified in the presence of uncertainties using with optimisation-based verification method. To the best of our knowledge, it is the first time that optimisation based verification process has been studied for this decision making collision avoidance algorithm.

To demonstrate the concept, a 6DOF UAVs model which was described in chapter 3 is used with a designed decision making collision avoidance algorithm. The uncertainties considered in the present analysis are the parameters representing the vehicle's mass and two aerodynamic coefficients. Moreover, the obstacle sensor data uncertainties and navigation sensor data uncertainties are considered in this study. Local and global optimisation-based verification methods are applied to the OAS and compared the results. Furthermore, in order to compare the performance and efficiency of the proposed optimisation-based verification approach for OAS, a Monte Carlo simulation is carried out.

7.2 Decision-Making Collision Avoidance Algorithm

This section presents a decision-making algorithm for pair-wise non-cooperative aircraft collision avoidance. Non-cooperative means that aircrafts do not collaborate in resolving the conflict, because either one of them is not equipped with sophisticated avionics, or communication between aircraft fails [111]. In general, one aircraft is considered as an intruder whereas the other one is assumed to be equipped with an ACA system, capable of detecting and avoiding the intruder without knowing its intentions.

7.2.1 Collision Geometry

The geometry which has been adopted for a collision situation between two aircrafts in the 3D North-East-Down (NED) reference frame is shown in Fig.7.1. The aircraft with ACA module (A/C_A) is considered as a point object with 3 Degree Of Freedom and velocity V_A, while the intruder (A/C_B) is modelled as a sphere with radius R (safety bubble) having velocity V_B. Given the described geometry, collision avoidance can be formulated as a two stage problem: First stage is Conflict detection*;* a potential conflict between two aircrafts will be detected. The second stage is Conflict resolution, determining if their future positions, after a certain amount of time should experience a loss of minimum separation. In such a case the trajectory of aircraft A/C_A has to be re-planned by solving a Conflict resolution problem.

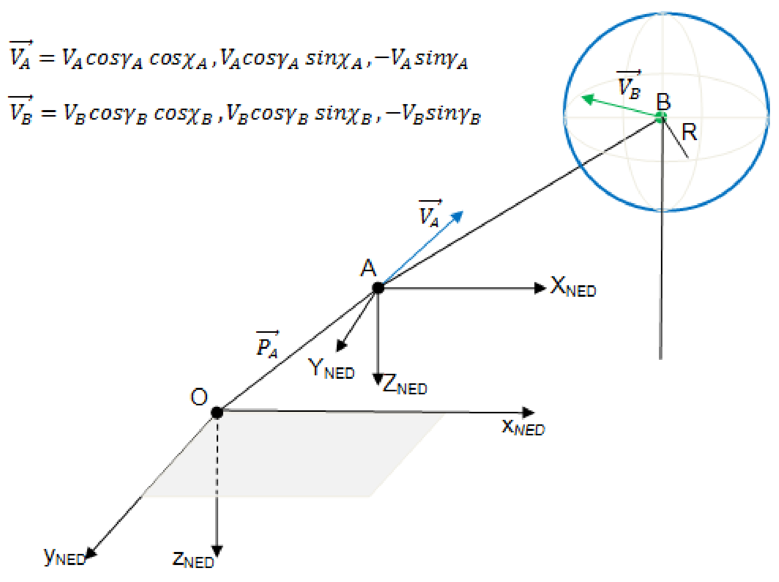

$\vec{V_A} = V_A\cos\gamma_A\cos\chi_A, V_A\cos\gamma_A\sin\chi_A, -V_A\sin\gamma_A$
$\vec{V_B} = V_B\cos\gamma_B\cos\chi_B, V_B\cos\gamma_B\sin\chi_B, -V_B\sin\gamma_B$
$\vec{V_B}$
B
R
$\vec{V_A}$
A
X_{NED}
$\vec{P_A}$
Y_{NED}
Z_{NED}
O
x_{NED}
y_{NED}
z_{NED}

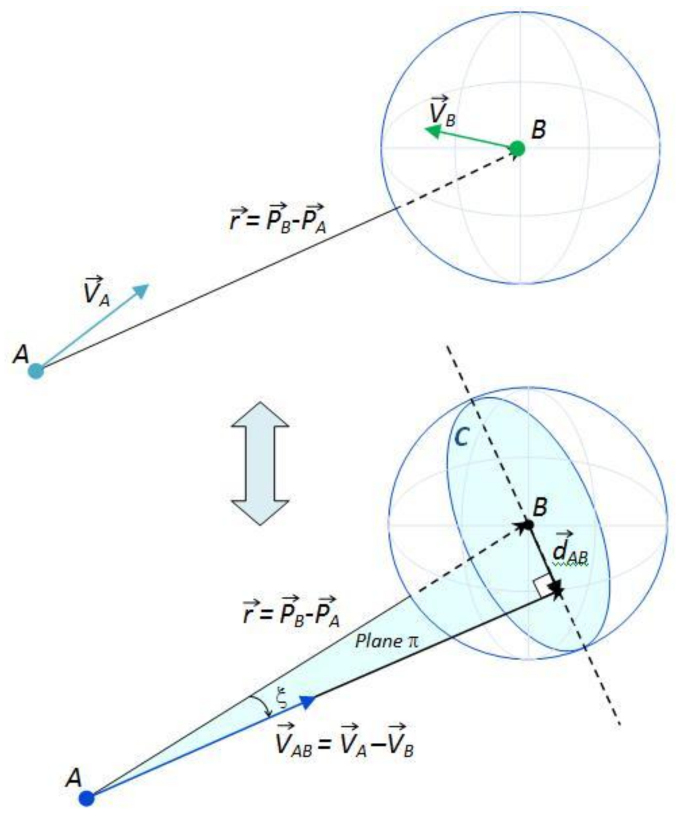
Figure 7.1: Collision Geometry between a point of mass A and a sphere B

7.2.2 Conflict Detection

The relative velocity vector $V_{AB}=V_A-V_B$ transforms a dynamic collision avoidance problem into a static problem: conflict geometry defined in Fig.7.2 is equivalent to a situation where sphere B is stationary and point of mass A moves with relative velocity V_{AB}. The plane π be the plane on which lie vectors $\vec{V}_{AB}$ and $\vec{r}$. This plane cuts sphere B determination a circle having radius R.

Let $\overrightarrow{d_{AB}}$ be a vector defined as the minimum separation distance experienced between aircraft, after certain time horizon. It can be calculated as follows:

$$\overrightarrow{d_{AB}} = \frac{\vec{r}.\overrightarrow{V_{AB}}}{\|\overrightarrow{V_{AB}}\|^2}\vec{V}_{AB} - \vec{r} \tag{7.1}$$



$\vec{V}_B$
B
$\vec{r} = \vec{P}_B - \vec{P}_A$
$\vec{V}_A$
A
C
B
$\vec{d}_{AB}$
$\vec{r} = \vec{P}_B - \vec{P}_A$
Plane π
ξ
$\vec{V}_{AB} = \vec{V}_A - \vec{V}_B$
A

Figure 7.2: Definition of minimum separation distance vector

Conflict detection stage is based on the following:

Theorem1: A point of mass A and a sphere B with radius R which are moving in a 3D environment with velocities V_A and V_B , respectively are headed for a collision if and only if the following conditions are satisfied:

$$\left\|\vec{d}_{AB}\right\| \leq R \text{ and } \dot{r} < 0 \tag{7.2}$$

7.2.3 Conflict resolution

The Conflict resolution strategy proposed in [111, 116], called minimum Deviation Control strategy is based on the analytical solution of the following kinematic optimisation problem. Find the minimum change in nominal trajectory of aircraft A to be forced (compatible with its envelope limitation and dynamic constraints) in order to avoid a collision with the safety bubble surrounding aircraft B.

Let $\overrightarrow{P_A}(t) = [x_A(t), y_A(t), z_A(t)]^T$ and $\overrightarrow{P_B}(t)$ be, respectively, A/C$_A$ and A/C$_B$ nominal trajectories. $\overrightarrow{P_A}(t)$ can be expressed in terms of velocity vector $\overrightarrow{V_A}(t)$ along the nominal trajectory as follows:

$$\overrightarrow{P_A}(t) = \overrightarrow{P_A}(t_0) + \int_{t_0}^{t} \overrightarrow{V_A}(\tau)\, d\tau \tag{7.3}$$

Let $\vec{P}_A^d(t)$ be the modified trajectory resulting from the demanded velocity vector function $\vec{P}_A^d(t) \equiv \vec{V}_A^d(V_A^d, \chi_A^d, \gamma_A^d, t)$. The deviation from the nominal trajectory of aircraft A is:

$$\vec{P}_A^d(t) - \vec{P}_A(t) = \int_{t_0}^{t} \left[\vec{V}_A^d(\tau) - \vec{V}_A(\tau)\right] d\tau \tag{7.4}$$

Minimizing nominal trajectory deviation-under envelope limitation and dynamic constraints-means minimizing the quantity $\left\|\Delta\vec{P}_A(t)\right\| = \left\|\int_{t_0}^{t} \Delta\vec{V}_A\,(\tau)d\tau\right\|$as stated by the following nonlinear programming problem:

$$\min_{V_A^d, \chi_A^d, \gamma_A^d} \left\|\int_{t_0}^{t} \Delta\vec{V}_A(\tau)d\tau\right\|$$

$$\text{s.t}\begin{cases} \left\|\vec{P}_A^d(t) - \vec{P}_B(t)\right\| \geq R,\ \forall \mathrm{t} \geq t_0 \\ V_A^d(t) \in [V_{A\,min},\ V_{A\,max}],\ \gamma_A^d(t) \in [\gamma_{A\,min}, \gamma_{A\,max}] \\ \exists t_a^A > 0 \end{cases} \tag{7.5}$$

Constraint (7.5)-1,

$$\left\|\vec{P}_A^d(t) - \vec{P}_B(t)\right\| = \left\|\vec{r}(t) + \int_{t_0}^{t} \Delta\vec{V}_A(\tau)d\tau\right\| \geq R$$

ensures that minimum separation distance R is never violated (Collision Avoidance); constraint (7.5)-2 represents the aircraft envelope limitations; constraint (7.5)-3 is a dynamic constraint, since the closed-loop systems “Aircraft & Autopilot” has a finite settling time t_a^A

for velocity vector changes, i.e., $\vec{V}_A^d$ cannot be reached instantaneously but requires a certain time t_a^A. In order to approach analytically the general Collision Avoidance problem (7.5), three assumptions are hereafter considered:

1. Change in velocity vector occurs only at time t_0 , i.e., $\Delta\vec{V}_A(t)$ is a step function
2. Straight aircraft trajectories at constant speeds and
3. No aircraft envelope limitations and dynamic constraints.

General problem (7.5), under assumptions (1)-(3), becomes

$$\min_{V_A^d,\chi_A^d,\gamma_A^d} \left\|\Delta\vec{V}_A\right\|$$

$$\text{s.t} \qquad \left\|\vec{d}_{AB}^d\right\| = R \tag{7.6}$$

It is to prove that this new problem admits an analytical solution, as follows [78]:

$$\vec{V}_A^d = \frac{V_{AB}\cos(\eta-\xi)}{sin\xi}\left[\sin\eta\,.\hat{V}_{AB} - \sin(\eta-\xi)\,.\hat{r}\right] + \vec{V}_B \tag{7.7}$$

where $\hat{r}$ and $\hat{V}_{AB}$ are respectively the unit vectors of $\vec{r}$ and $\vec{V}_{AB}$; moreover, $\eta = sin^{-1}\frac{R}{\|\vec{r}\|}$ and it has the same sign of ξ, angle formed by vectors $\vec{r}$ and $\vec{V}_{AB}$. General problem (7.5) has been simplified thus the minimisation of A/C$_A$ nominal trajectory deviation has been made equivalent to the minimisation of vector $\Delta\vec{V}_A$. This new problem admits analytical solution (7.7) to the collision avoidance problem and does not require the resolution of any numerical optimisation problem, thus resulting suitable for real-time applications. Collision avoidance manoeuvres are performed in 3D by changing simultaneously aircraft speed module, track and slope angles.

7.3 Autonomous Collision Avoidance Systems

The closed-loop control system of ACA module is shown in Fig.7.3, where it is represented by a decision-making algorithm, having as input speed and position of own aircraft (P_A , V_A) and the intruder (P_B , V_B). The outputs of the decision-making algorithm are reference signals to the autopilot, in terms of demanded speed module (V_d), slope angle (γ_d) and track angle (χ_d).

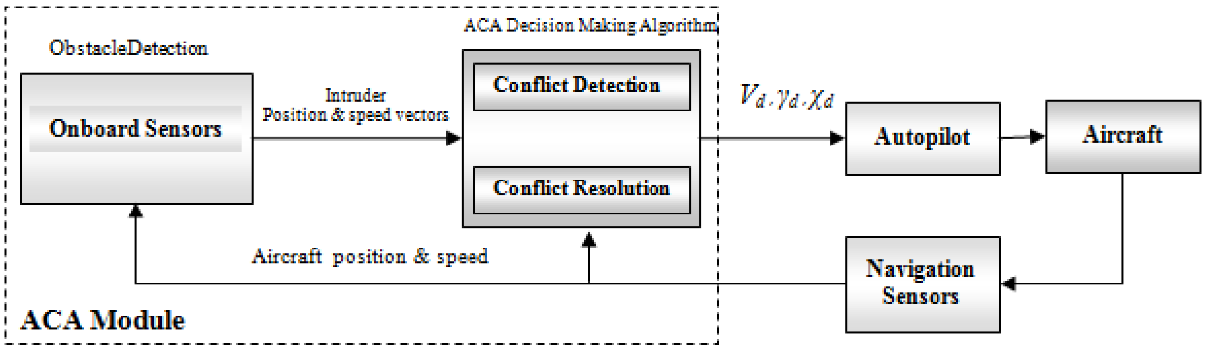


Figure 7.3: Autonomous collision avoidance control systems

Based on a 6 Degree of Freedom (6DoF) kinematic and dynamic model of a UAV with the decision making collision avoidance algorithm is developed in 3D space. Recall the kinematic and dynamic equations of UAVs which are described in the chapter 3, section 3.4.1 and 3.4.2. The configuration vectors = $[x, y, z, \phi, \theta, \psi]^T$ is used to specify the position and orientation of the UAV in the global coordination, where $q_o=[x, y, z]^T$ is the c.g. (center of gravity) position of the vehicle and $\boldsymbol{\varphi} = [\phi, \theta, \psi]^T$ are the Euler angles, with ϕ as the roll, $\theta = \gamma$ as the pitch, and $\psi = \chi$ as the yaw. After the desired global velocity is calculated by the Eq.7.7, the corresponding desired linear velocity $u_d=V_d$ and attitude $\boldsymbol{\varphi}_d = (\phi_d , \theta_d=\gamma_d , \psi_d=\chi_d)$ can also be obtained based on UAV's kinematic model using the equations of 3.45-3.49. Four PID controllers are designed for controlling linear velocity and three attitude angles using the Eq.3.50.

7.4 Collision Avoidance Simulation at Nominal case

Algorithm verification at nominal parameters has been carried out via numerical simulations by defining the collision scenarios with and without decision making collision avoidance algorithm. The Eq.7.2 offers an analytical criterion to find initial positions and speed vectors of the two aircrafts which cause initial relative velocity vector to enter in the safety bubble. By considering this conflict scenario, the collision geometries approach is reduced the chances of collisions. The nominal parameter values are *m=1.9 kg* and $C_{m\delta e}$ *=-1.13*. The initial linear and angular velocity vectors for UAV and intruder are chosen as (20, *0, 0) m/s* and *(0, 0,0) rad/s* for the nominal case. The initial Euler angle for UAV and intruder are *(0, 0, 0.9) rad*. The PID controller gains are tuned and set to fixed values for the verification process. In the simulation, the UAV starts from *(0, 0, 0)m*, and the intruder‘s initial starting point is *(0, 800, 0)m* with a safety radius *R* of 5*0m.*

The simulation results for UAVs collision avoidance are presented at the nominal parameters. First the simulation result without decision making collision avoidance algorithm at 32 seconds is shown in Fig.7.4. In this case, the minimum distance to the obstacle is obtained as 4.49m and it can guarantee a collision if collision avoidance manoeuvre is not performed. After that, the simulation results with decision making collision avoidance algorithm at 30, 35 and 50 seconds are shown in Figs.7.5–7.7. The minimum distance to the obstacle at nominal parameters obtained from this simulation scenario is *78.11m* which is greater than the safety radius of *50m* (d_{min}> R). i.e. no collisions occurred. It means that the avoidance manoeuvres skim the safety bubble.

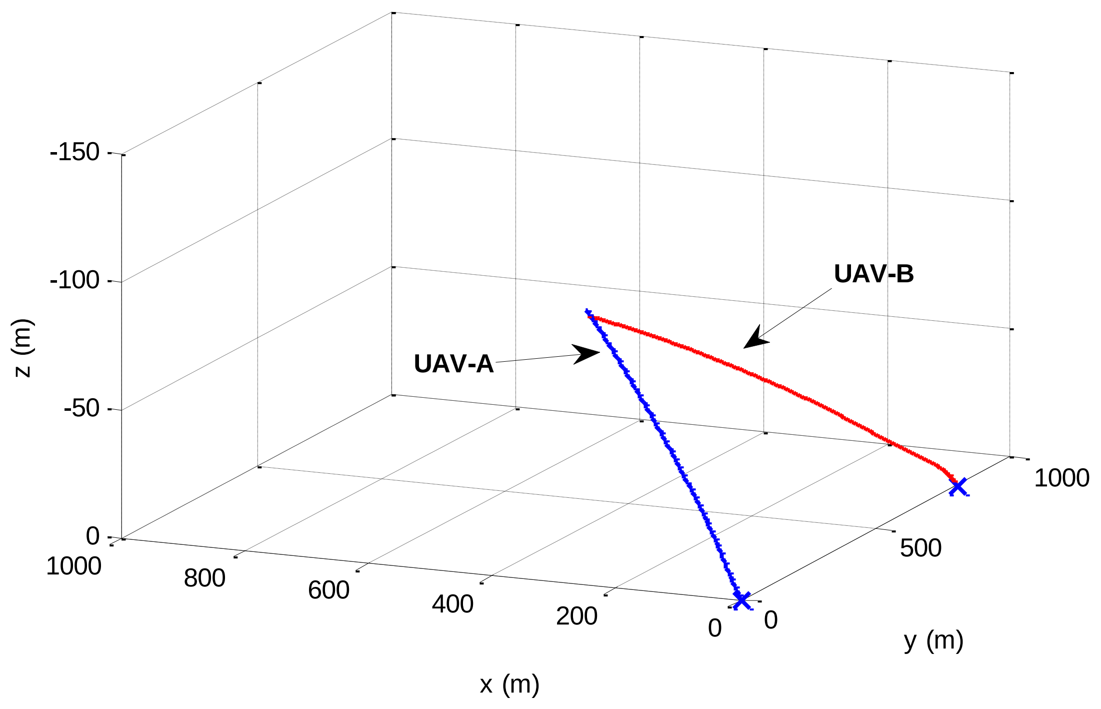


Figure 7.4: Simulation response without decision making collision avoidance algorithm at t=32 sec

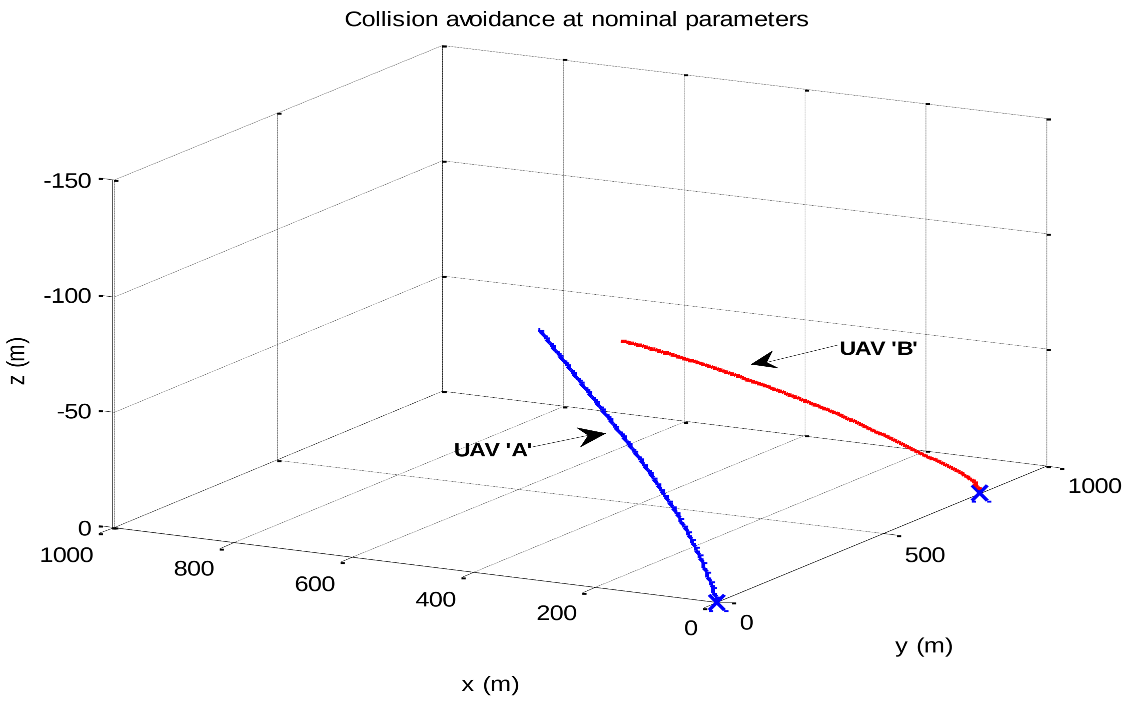


Figure 7.5: Simulation response with decision making collision avoidance algorithm at t=30 sec

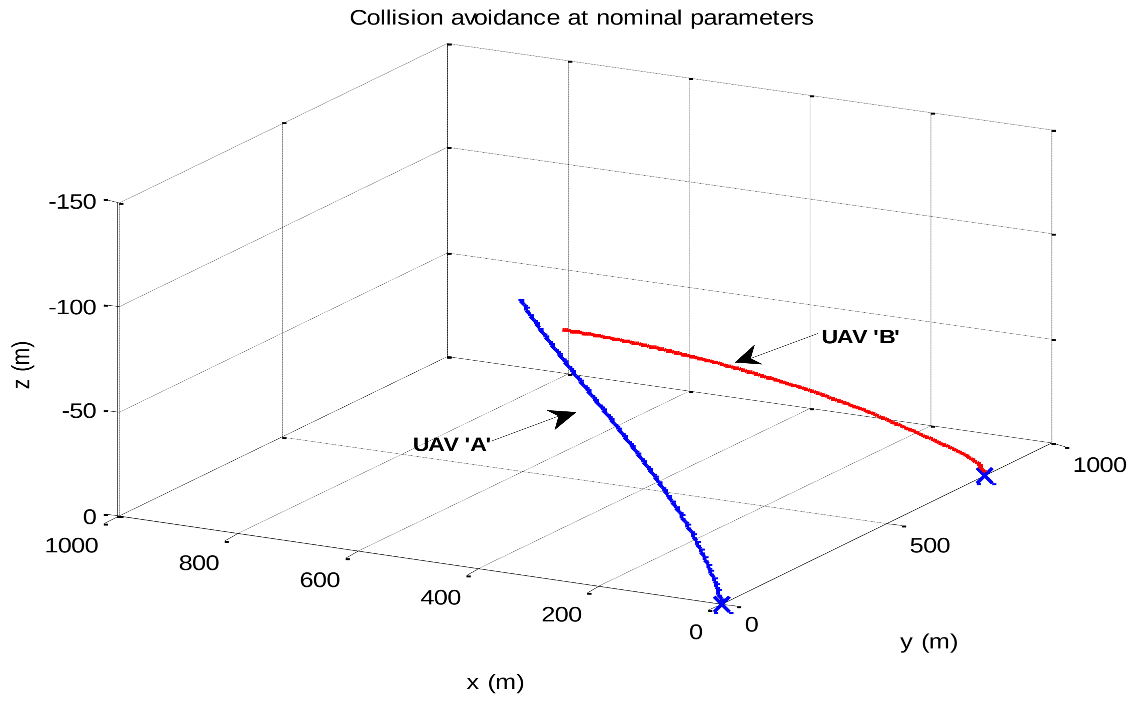


Figure 7.6: Simulation response with decision making collision avoidance algorithm at t=35 sec

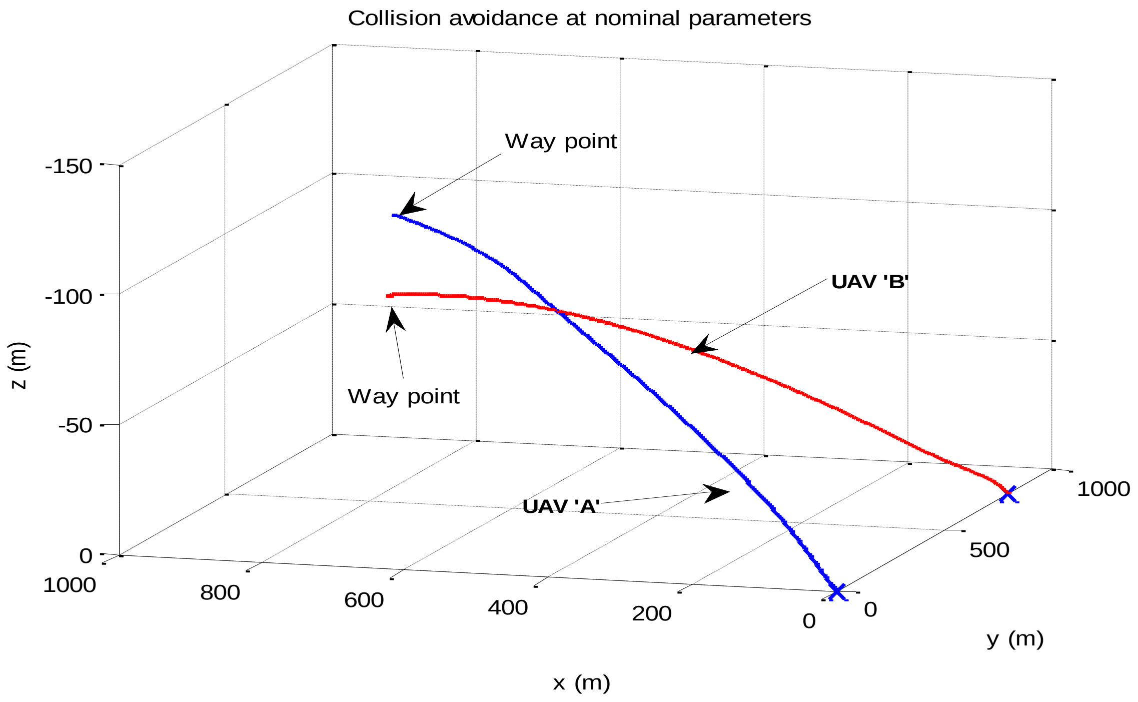


Figure 7.7: Simulation response with decision making collision avoidance algorithm at t=50 sec

7.5 Initial Robustness Analysis for ACA System

Initial robustness analysis of the proposed decision making collision avoidance algorithm is carried out in this Section. Uncertainties are introduced in navigation sensor data as follows: UAV velocity: $u_A = u + \Delta u$; UAV track angle: $\psi = \psi + \Delta\psi$; Uncertainties are introduced in obstacle detection sensor data as follows: Intruder velocity: $u_B = u_0 + \Delta u_0$; Intruder track angle: $\psi_B = \psi_0 + \Delta\psi_0$. where u and ψ are UAV velocity and track angle readings at the nominal case respectively; u_0 and ψ_0 are Intruder velocity and track angle at the nominal case respectively. Δu, $\Delta\psi$, Δu_0 and $\Delta\psi_0$ are sensor data errors in u, ψ ,u_0 and ψ_0 respectively. These uncertainties are chosen within the bounds to find the worst-case condition. u and u_0 are considered within the bounds of [-2, 2] ms^{-1} and ψ and ψ_0 are set to [-0.2, 0.2] *rad*.

Uncertainties are considered in the dynamic model (mass and aerodynamic coefficients), and each uncertain parameter is allowed to vary within ± (20)% of its nominal value. These are firstly considered within lower and upper bounds, i.e. $m = [1.52, 2.28]\ kg$ and $C_{m\delta e} = [-1.356, -0.904]$. For the purpose of comparison, these uncertain parameters are normalized to have a variation within the range. Fig.7.8 shows variations of the minimum distance to the obstacle with respect to the normalized uncertain parameters of mass and $C_{m\delta e}$. There is a significant variation in the distance with the variations of these uncertain parameters. The minimum distance to the obstacle monotonically decreases with the increase of the mass m while d_{min} increases with the increase of $C_{m\delta e}$. Fig.7.9 and 7.10 show the variations of the minimum distance to the obstacle with respect to the sensor data variations. It clearly shows that for different uncertain parameters, the different influences on the minimum distance to the obstacle are found. The d_{min} decreases with the increase of UAV's velocity and track angle while d_{min} increases with the of intruder's velocity.

The UAV model used for this case study is same as used in case studies 3 and 4. The controller gains are tuned and set to fixed values, i.e. same controller gains are used in these three case studies. However, different collision avoidance algorithms are used in the development of OAS. Therefore, different motion planner parameters are tuned for these three case studies. Figs.3.32, 3.34 and 7.8 show variations of the minimum distance to the obstacle with respect to the normalized uncertain parameters. It clearly demonstrates that the safety of the vehicle is dependent on the proposed control laws and also collision avoidance algorithms.

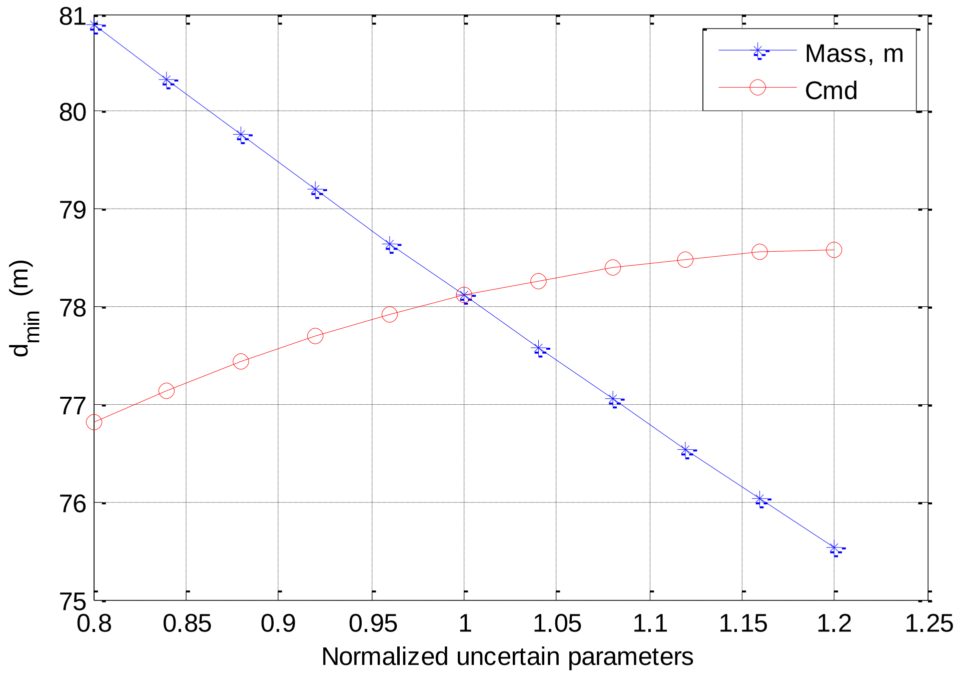


Figure 7.8: Mass and $C_{m\delta e}$ variations

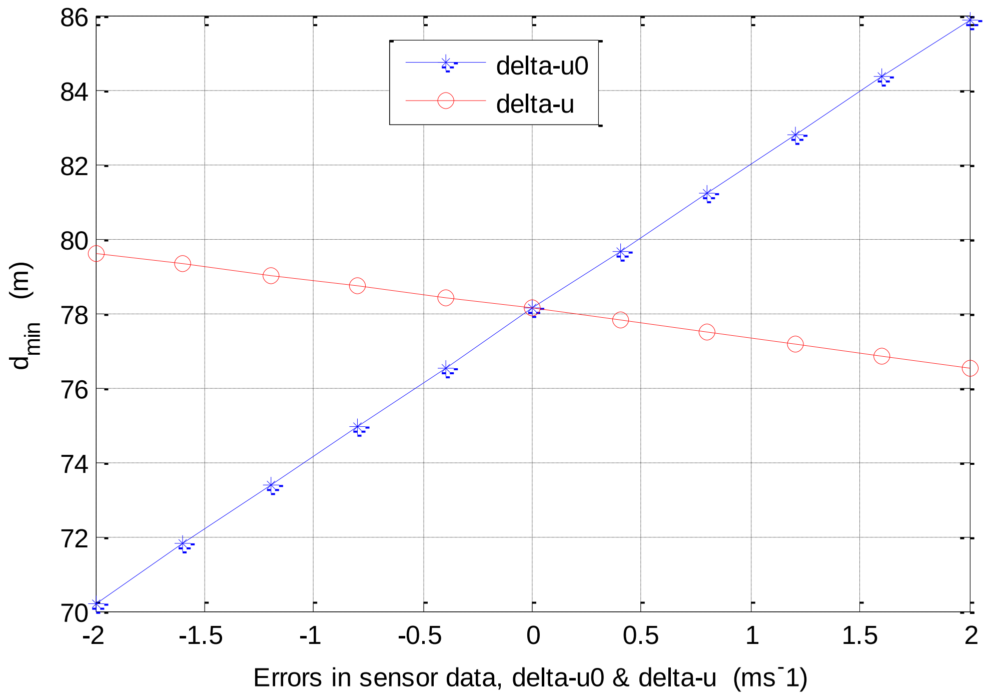


Figure 7.9: Sensor data Δu and Δu_0 variations

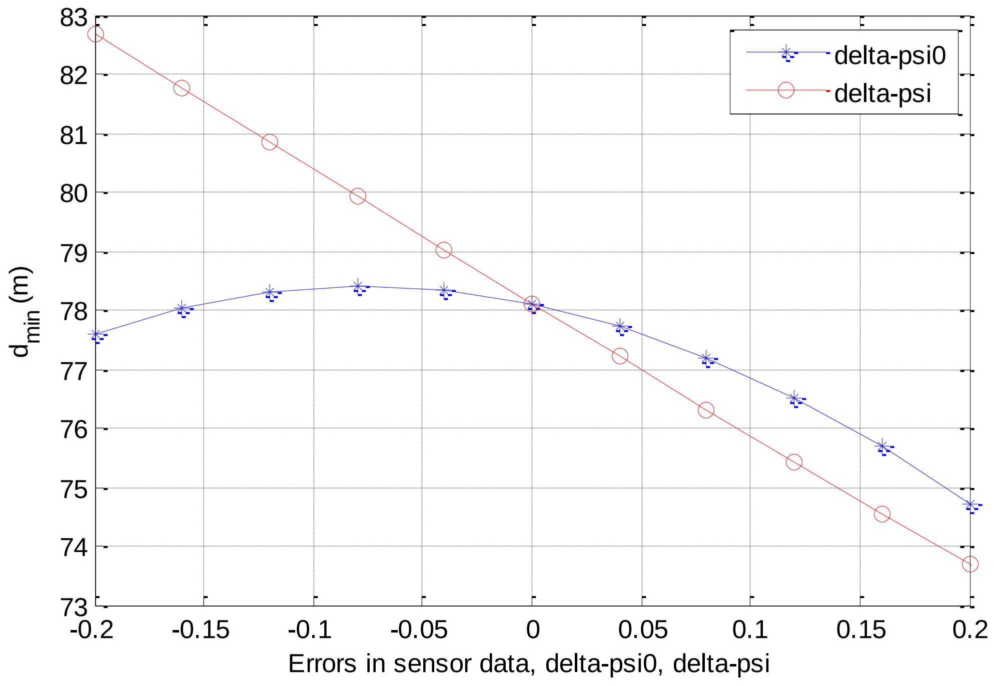


Figure 7.10: Sensor data $\Delta\psi$ and $\Delta\psi_0$ variations

7.6 Optimisation-based worst-case analysis Approach

This section describes the results obtained by applying the optimisation methods presented in the previous section of ACA system for searching of worst-case manoeuvres. The search for worst-case collision scenarios is performed by using local and global optimisation algorithms over a parameter space. The worst-case search methods apply for nonlinear programming techniques to minimize suitably defined clearance criterion to determine worst-case combinations of uncertain parameters. The objective function in the optimisation is chosen as the minimum distance from the vehicle to the obstacle during the manoeuvre which is defined in Eq.7.8 as follows:

$$\begin{aligned} d_{min} &= min(d(t)) \quad \text{for } \ t \leq T\ (sec) \\ s.t \quad & P_L \leq P \leq P_U \end{aligned} \tag{7.8}$$

where P is the uncertain parameters set; P_L and P_U are lower and upper bounds of P; T is the time period of the collision avoidance manoeuvre; *d(t)* is the distance to the obstacle and is calculated using simulation with the completed model of the vehicle in Fig 7.3.

Table 7.1 summarize the results for implementation of the local search method of *fmincon* algorithm. By studying the table, objective function value at case-1, *fmincon* converges to the minimum distance to the obstacle of 58.97m while it is 65.36m at case-2. There are huge differences in the converging worst-case condition. Therefore, the results clearly show that *fmincon* does not give the same solutions with the different starting points. Local optimisation-based method is not suitable for this study. Because of this worst-case violation of the optimal solution, the global optimisation methods are considered to find the true worst-case. As expected, the global methods are generally more expensive to use than local methods in terms of necessary number of function evaluations.

TABLE.7.1 LOCAL OPTIMISATION RESULTS FOR UAV ACA SYATEM

Algorithm *fmincon*	Starting point $[m, C_{m\delta e}, \Delta u, \Delta\psi, \Delta u_0, \Delta\psi_0]$	Convergent point $[m, C_{m\delta e}, \Delta u, \Delta\psi, \Delta u_0, \Delta\psi_0]$	d_{min}(m)
Case 1	[1.71, -1.13, 1, 0.1, -1, -0.1]	[2.28, -1.36, 1.99, 0.2, -2, -0.2]	58.97
Case 2	[2.09, -1.017,1,0.1, 1, 0.1]	[1.52, -1.36,-2,0.2,-2, 0.2]	**65.36**

The results for worst-case condition and parameters determination obtained using the global methods GA, GLOBAL and DIRECT are also presented in this section. Fig.7.11 shows the GA run with the population size=20 and crossover fraction=0.8. The GA optimisation is terminated after given iterations (100). The GLOBAL optimisation with UNIRANDI local search method is applied to find the global solution for the ACA system. The DIRECT algorithm terminates as soon as it exceeds the given iterations of 500. DIRECT iteration history of Fig.7.12 shows that the minimum distance to the obstacle falls rapidly in the beginning, going below 58m after 30 iterations.

A comparison of the minimum distance to obstacle before and after the optimisation is given in Table.7.2. A significant change in minimum distance to obstacle is seen after the optimisation. All optimisation algorithms are performed in MATLAB 2012b and Intel (R) Core(TM) 2 Duo CPU (3.16GHz). The minimum distance to the obstacle obtained from

DIRECT, GA and GLOBAL algorithms are almost same. GLOBAL took 5425 functions evaluation with 300 sampling points and 13 local minima are found. GLOBAL took 1 h 11 minutes to converge to the global minimum. DIRECT took 31697 function evaluations and it took 10 hours 27 minutes to converge to the global minimum. GA took only 37 minutes to converge to the global minimum and 2020 function evaluations are taken. GA performs faster than other two algorithms; however, DIRECT algorithm can guarantee the global minimum. Final values of six design variables after optimisation are shown in Table.7.3. All three global algorithms are converged to nearly same values.

TABLE.7.2. COMPARISON OF WORST-CASE CONDITION, d_{min}(m)

Before optimisation d_{min} (m)	After Optimisation d_{min} (m)		
Norminal Case	**DIRECT**	**GLOBAL**	**GA**
78.11	56.63	56.63	56.64

TABLE.7.3. COMPARISON OF WORST-CASE PARAMETERS VALUES

Design Variable	Initial Value	Final Value		
		DIRECT	**GLOBAL**	**GA**
m	1.9	2.28	2.28	2.28
$C_{m\delta e}$	-1.13	-1.356	-1.356	-1.356
Δu	0	1.9997	2.0	2.0
$\Delta\psi$	0	0.2	0.2	0.2
Δu_0	0	-1.9997	-2.0	-1.999
$\Delta\psi_0$	0	0.2	0.2	0.1999

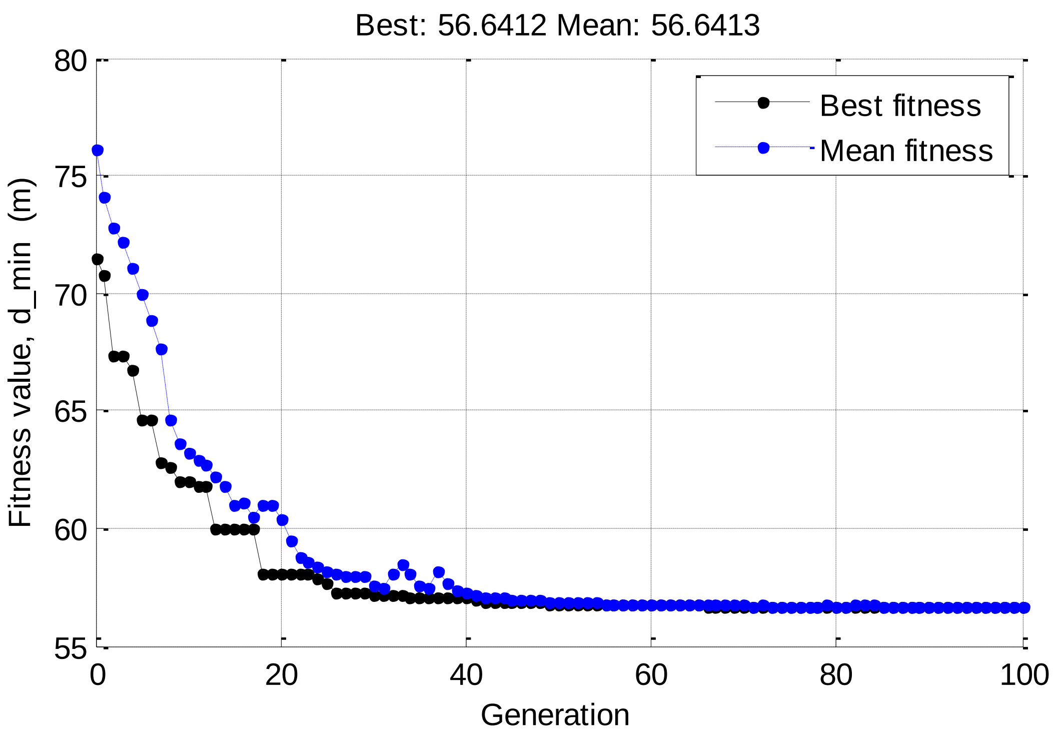


Figure 7.11: GA- No of generations vs. fitness value

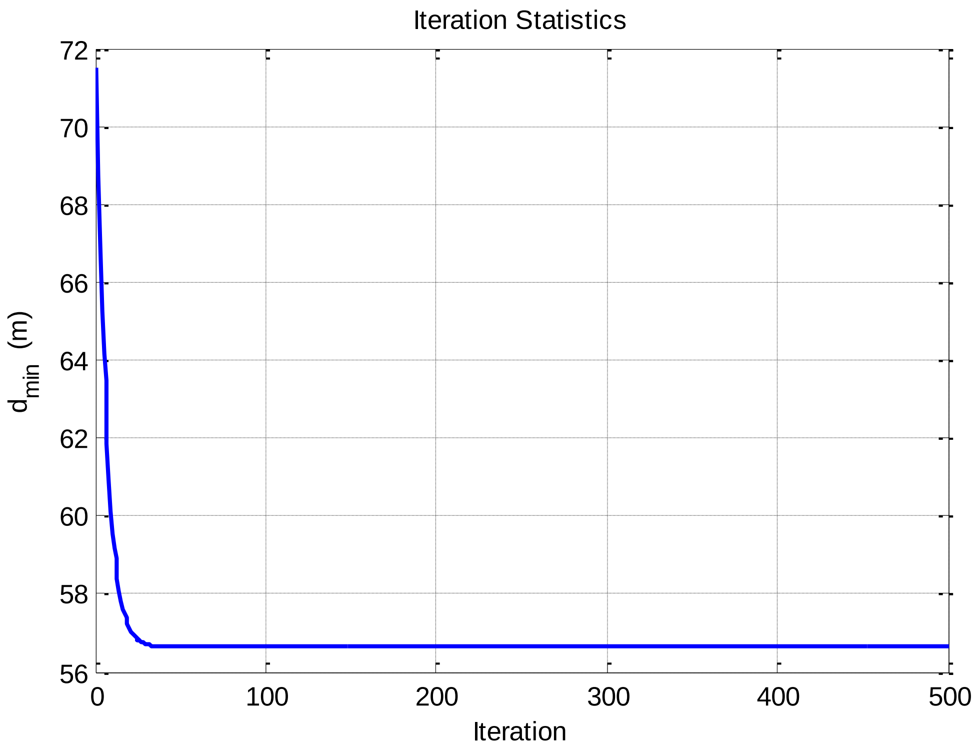


Figure 7.12: *DIRECT* algorithm- Iteration vs. Fitness value

Of the presented global optimisation-based verification method, the optimized minimum distance to the obstacle d_{min} is decreased from 78.11 to 56.63m which is greater than the specified safety radius of *R (50m).* The algorithm has proved its validity in considered scenarios, performing avoidance manoeuvres. The performance of the decision making algorithm at worst-case parameters is checked with simulation response at 35 sec which is shown in Fig.7.13. This concludes that the decision making collision avoidance algorithm and the controller provide adequate performance at the worst-case parameters. Furthermore, in the presence of all the described variations and uncertainties, the safety margin for anti-collision is respected. The time versus distance to the obstacle at the nominal and worst-case parameters is shown in Fig.7.14. It clearly shows that there is a significant difference in the minimum distance to the obstacle at nominal and worst-case parameters during the manoeuvre.

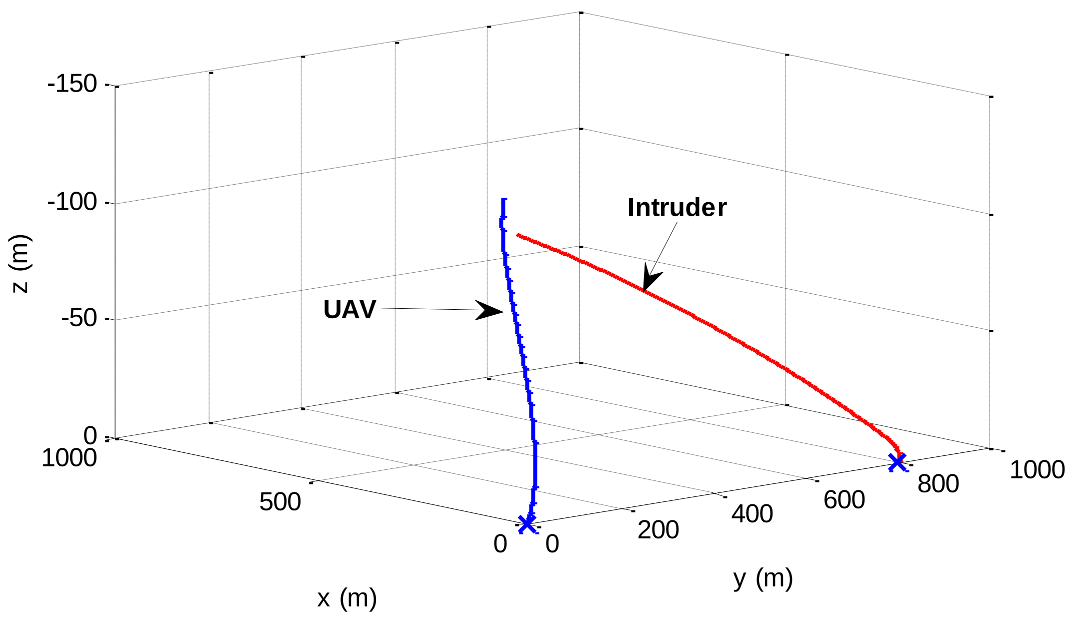


Figure 7.13: Simulation result for ACA system at worst-case parameters at 35 sec.

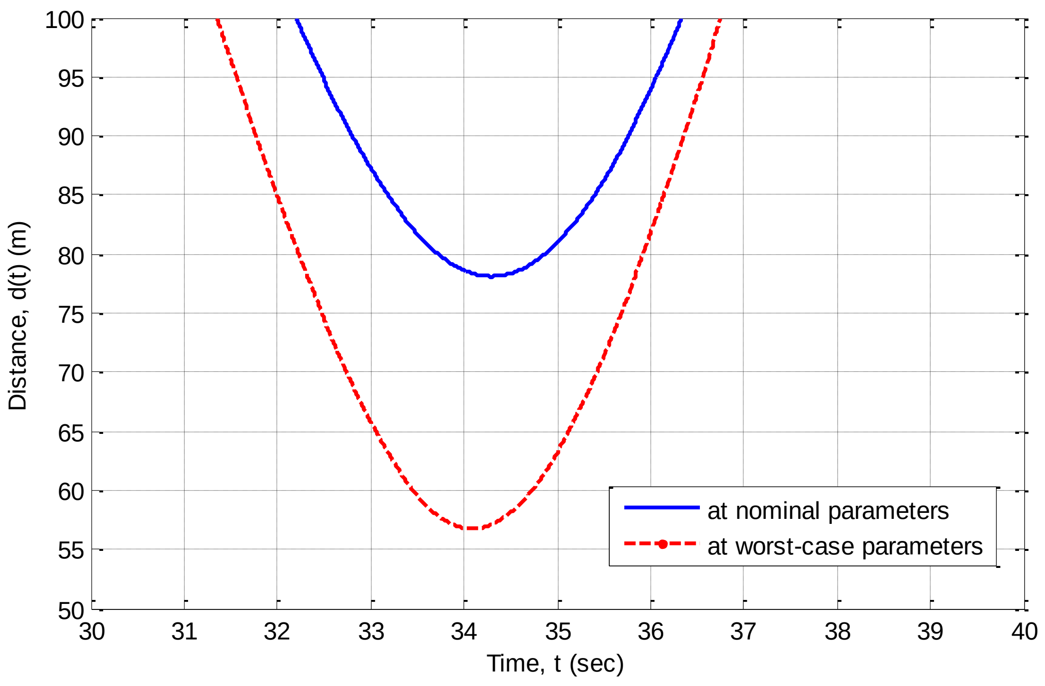


Figure 7.14: Time vs distance to the obstacle at nominal and worst case parameters

7.7 Monte Carlo Simulations

To verify the proposed worst case analysis methods and benchmark their performance, the most widely used Monte Carlo method (MCM) is applied to the ACA system. All worst-case search parameters are assumed to be uniformly distributed over the defined intervals. Monte Carlo simulation is executed with 5,000 runs to find the worst case scenario and the result is shown in Fig.7.15. The minimum distance to the obstacle d_{min} at the worst case obtained by MCM is 60.1899m while that identified by the optimisation based automatic search methods presented in this study is 56.63m. We can conclude that unsatisfactory worst-case value is detected from Monte Carlo method, and proposed optimisation based verification method can demonstrate the level of confidence of true worst-case from a series of optimisation runs.

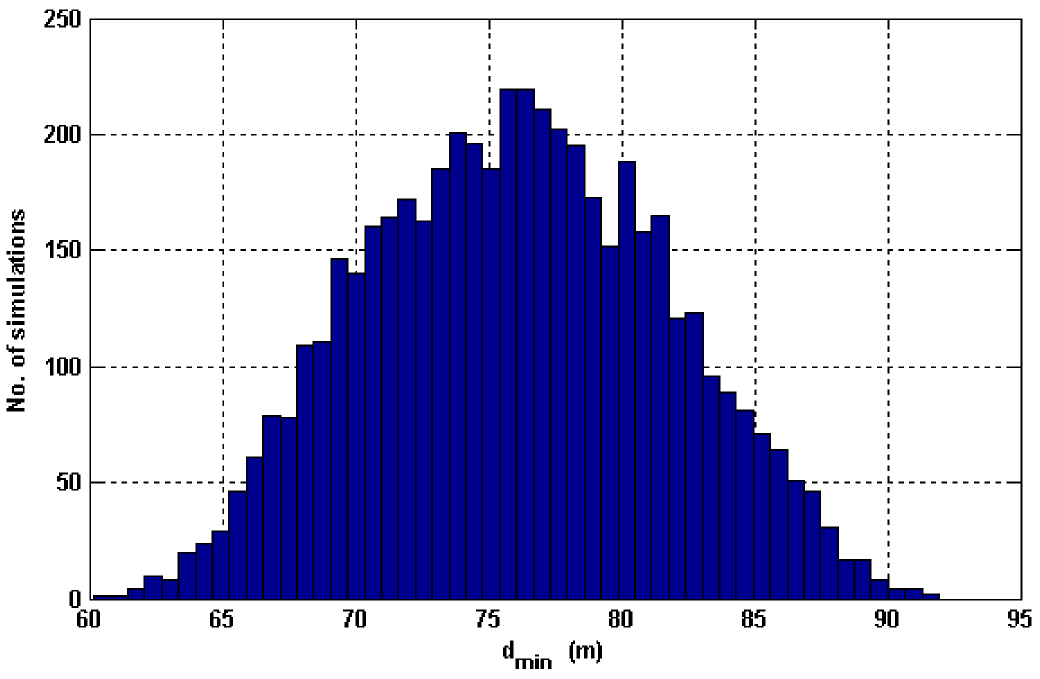


Figure 7.15: Monte Carlo simulations results for ACA system

7.8 Conclusion

This chapter described the use of optimisation-based verification for analyzing nonlinear robustness analysis of decision making collision avoidance algorithm. The decision making collision avoidance algorithm for pair wise non-cooperative aircraft, having the capability of avoiding a safety bubble is described in this chapter. The proposed solution for the collision avoidance problem here considered is derived on the basis of a 3-dimensional analytical approach with the simultaneous change of all control variables of speed module, track and slope angles. The effectiveness of the algorithm here described is proved by number of simulations. Uncertainties are considered in the dynamic model and on-board and navigation sensor data within the lower and upper bounds. Local and global optimisation methods are applied to the ACA system, and the results of local search method are violated with different starting points. Global optimisation algorithms are converged to the same global optima. Furthermore, the worst case condition obtained from the MCM is not the true worst case and there is a high chance of missing the true worst case solution in this approach. Therefore, the major conclusion from the verification results achieved is that optimisation-based worst-case search proved to be a general, direct and reliable approach to solve clearance problem.

Chapter 8

Conclusions and Future Work

8.1 Conclusions

In this thesis, new verification method has been developed for the clearance of collision avoidance system for unmanned vehicles. Verification of safety-critical system must be performed to prove that the controlled vehicles meet all clearance criteria.

Overall goal: *To develop a method to verify the safety of collision avoidance system for unmanned vehicles.*

In order to reduce the risk of collisions in the presence of all possible parameter variations, extensive computer aided simulations and robustness analysis were performed in this thesis. In developing optimisation-based worst-case analysis for verification of collision avoidance algorithms, the minimum distance to the obstacle during collision avoidance manoeuvre is defined as the cost function in the time domain. The worst-case search method aims to find all the possible worst cases in order to verify the collision avoidance algorithms in the presence of all possible uncertain parameters bounds. This requires an optimisation algorithm that converges to the global optimal solution. This verification of OAS becomes a very expensive and time consuming task as the collision avoidance algorithms and control systems become more complex.

Two existing different collision avoidance algorithms which are potential field method and decision making algorithm were developed and verified these algorithms within the uncertain parameters bounds using with optimisation-based verification approach. The verification technique proposed in this paper may be applicable for other moving obstacle avoidance algorithms after appropriate modifications. Therefore, the strength of the optimisation-based verification approach is its flexibility in that it can be used to check all linear or nonlinear clearance criteria for all collision avoidance algorithms. Different optimisation methods, such as gradient-based local optimisation (Sequential Quadratic Programming), two stochastic global optimisation methods (Genetic algorithms and GLOBAL algorithm), a deterministic global optimisation algorithm (DIviding RECTangles) and finally hybrid optimisation approach were applied to the OAS to find the worst case scenarios. The main challenges of solving the worst case distance are the presence of local minima. Four benchmark case studies were presented using with potential field method.

Firstly, kinematic and dynamic equations of the simple unicycle robot were presented and controller was chosen based on these equations. The inner-outer-loop control architecture is used where the inner-loop controller is a PID controller. A local planner in the outer-loop was developed using the artificial potential field method. Secondly, more complex pioneer 3-DX robot was presented and moving obstacle avoidance algorithm were developed using with potential field method. Parametric uncertainties, sensor uncertainties and structural mismatching between the model used for the control and collision avoidance algorithm design and the real vehicle have been addressed. Eight uncertain parameters including the changes of mass, inertia, friction coefficients, side slip and sensor data are considered in this case study. Thirdly, a 6DOF UAV model was used in the case study with a designed static collision avoidance algorithm, and four PID controllers are designed for controlling linear velocity and three attitude angles. Mass and two aerodynamic coefficients variations were considered for the verification purpose. Next, this study was extended to verification of moving OAS for UAVs using with potential field method.

Then an optimisation-based approach was developed to find the worst cases which are defined by the minimum distance to the obstacle in the presence of all possible described parameters variations. For local optimisation methods, different worst cases have been identified when the optimisation started from different initial conditions. The local optimisation does not give the unique solution for four case studies because it is a non-convex nonlinear optimisation problem and it is possible to miss the worst cases. To overcome this problem, stochastic global optimisation algorithms including GA and GLOBAL methods were applied to the problem of analyzing the robustness properties of a vehicle dynamic system. However, as both are stochastic global optimisation algorithms and cannot guarantee the optimisation process converges to the global solutions, i.e. the worst-cases. To overcome this drawback a deterministic global optimisation algorithms, DIRECT method has been investigated for the worst-case analysis. Compared with other global optimisation algorithms in this study, DIRECT algorithm can guarantee the worst cases are found. Moreover, in order to increase the efficiency of the results, the H-DIRECT algorithm has been investigated. The results show that it provides a most promising candidate for the optimisation-based verification process. Therefore, the presented collision avoidance algorithms and controllers function correctly in the presence of parameters variations.

Furthermore, the Monte Carlo simulations were carried out to compare the proposed optimisation verification method. Monte Carlo method is a widely used methods and industrial practice to identify the worst cases. The worst case condition obtained from the MCM for case study-2 is not the true worst case and there is a high chance of missing the true worst case solution in this approach.

Finally, a verification of decision-making algorithm for pair wise non-cooperative aircraft collision avoidance has been presented. The decision-making algorithm for aircrafts collision avoidance, having the capability of avoiding a safety bubble has been described. Based on 6 DOF kinematic and dynamic model of a UAV, decision-making collision avoidance algorithms were developed in 3D space. Four PID controllers were designed for controlling linear velocity and three attitude angles. Uncertainties were introduced in navigation sensor data and onboard obstacle detection sensor data. Six uncertainties were considered in this case study and compared the results. Of the presented optimisation algorithms, the local optimisation depends on the smoothness of the objective function. Therefore, it is difficult to select the true worst case for non-convex problem. The all global optimisation algorithm presented in this thesis were performed well for this case study. However, the DIRECT method has the proof of convergence. There is most significant different in the minimum distance to the obstacle at nominal and worst case parameters. The results obtained in this thesis clearly demonstrate the overall goal of this project.

8.2 Future work

The optimisation-based verification for OAS algorithm proposed in this thesis can be expanded in future work to support other complex scenarios. Many real-time applications may contain more complicated scenarios of collision avoidance problems. Therefore, clearance criterion of minimum distance to the obstacles will be identified for more complex scenarios within the uncertain parameters bounds and any failure conditions. In order to verify the complicated collision avoidance scenarios, multiple design objectives need to be checked and analysed simultaneously. Only single objective function was considered in this thesis, and this work will be extended to the multi-objective optimisation problem to find the worst case conditions for complicated static and moving collision avoidance problem. Multi-objective evolutionary optimisation algorithm such as MOGA presented in Ref [119], and multi-objective optimisation method for global search using DIRECT and GA algorithm introduced and discussed in Ref [120] would seem to provide obvious choices to extend the proposed approach to complicated scenarios.

The global optimisation algorithms are generally more expensive to use in terms of necessary number of function evaluations and computation time. The main industrial benefits of new methods should be related to reducing the involved effort and cost, while getting sufficiently reliable results, or increasing the reliability of the analysis results within a reasonable amount of effort. Safety and reliability is a significant challenge for current safety-critical system. Therefore, this proposed optimisation-based verification algorithm for OAS will be further analysed to reduce the cost and time.

The decision-making algorithm discussed in chapter 7 will further include the consideration of all the Right-of-Way rules in planning the collision avoidance manoeuvre, in such a way as this manoeuvre will be fully compliant with Visual Flight Rules. After considering these Right-of-Way rules, the optimisation-based verification method will be applied and analyzed the worst cases.

This optimisation approach can be applied to extremely complex nonlinear vehicle simulation models and analyzed the worst cases. And also, the verification process of OAS can be applied to a complete closed-loop control system model to simulate various collision scenarios operated by pilot in manual mode. More complex models are necessary to execute complex manoeuvres, as for example, those necessary to evaluate protection laws violation criteria or recovery manoeuvres from failure cases [117]. Therefore, the development of human pilot models with realistic biomechanical features is an important research challenge for the clearance of safety-critical systems [117].

This verification approach provides much useful information for example worst-case parameters combinations which can serve to increase the performance of the collision avoidance system or to redesign the flight control laws and collision avoidance algorithms. Therefore, clearance of OAS would potentially contribute to reduce global costs for collision avoidance algorithm testing, controller tuning and assessment.

The sum of squares - SOS programming can be applied to the OAS. The approach is applicable to nonlinear systems described by polynomial dynamics and it relies on connections between SOS polynomials and positive semidefinite matrices. Parrilo [118] proposed the computational tools for estimating regions of attractions, reachability sets, input-output gains, and robustness with respect to uncertainty. There are two keys in this approach. First, sufficient conditions for many nonlinear analysis problems can be formulated as set containment conditions involving either a Lyapunov function or a storage function. Second, the set of containment conditions can be reformulated as polynomial non-negative conditions using a generalized version of the S-procedure. These tools can be used to provide additional confidence when validating the performance of a flight control laws and collision avoidance algorithms.

Loss of control (LOS) remains one of the largest contributors to aircraft fatal accidents worldwide [117]. Research is underway at the National Aeronautics and Space Administration (NASA) in the development of advanced onboard system technologies for preventing or recovering from loss of vehicle control and for assuring safe operation under off-nominal conditions associated with aircraft LOS accidents. Future aircraft control systems will be expected to provide resilience under off-nominal conditions operate as a component of a larger resilient flight system. The broader resilient flight system will include vehicle health management, flight safety management and reliable crew interface management functions. V&V technologies must also be developed and applied to these technology areas for an improved understanding of safe and unsafe regions of operation under off-nominal conditions, and for the ultimate certification of these technologies.

Appendix A

Publications

A.1 Published paper

1. S. Srikanthakumar, W. H. Chen. Worst-case analysis of moving obstacle avoidance systems for unmanned vehicles. Robotica, Cambridge University Press 2014, doi: 10.1017/S0263574714000642.

2. S. Srikanthakumar, C. Liu, W. H. Chen. Optimization-based safety analysis of obstacles avoidance systems for unmanned aerial vehicles, Journal of Intelligent and Robotic Systems, Vol. 65, Issue. 1-4, pp. 219-231, January 2012.

3. S. Srikanthakumar, W. H. Chen. Optimisation-based clearance process of obstacle avoidance systems for unicycle-like mobile robot. International Journal of Automation and Computing, Vol. 8, Issue. 3, pp. 340-347, August 2011.

4. S. Srikanthakumar, C. Liu, W. H. Chen, Clearance process of obstacle avoidance systems for unmanned aerial vehicles, 4th European Conference for Aerospace Sciences, EUCASS, Saint Petersburg, Russia, 4-8 July 2011.

5. S. Srikanthakumar, W. H. Chen. Optimization-based safety analysis of obstacles avoidance systems for unmanned aerial vehicles, International Conference on unmanned aircraft systems, ICUAS, Denver, USA, 24-27 May 2011.

6. S. Srikanthakumar, W. H. Chen. Optimisation-based clearance process of obstacle avoidance systems. International Conference on Automation and Computing, Birmingham, 11th September 2010.

References

[1] C. Fielding, A. Varga, S. Bennani, and M. Selier, editors, *Advanced techniques for clearance of flight control laws*. NO. 283 in Lecture notes in Control and information sciences (LNCIS.283), Springer, Verlag, 2002.

[2] C. Fielding, "The Design of Fly-by-wire Flight Control Systems", BAE systems (Operations) Limited, 2000.

[3] G. S. Tallant, J. M. Buffington, V. W. Crum, B. Krogh, C. Plaisted, R. Prasanth, P. Bose and T. Johnson, "Validation & Verification of Intelligent and Adaptive Control Systems", AIAA, Guidance, Navigation, and Control Conference and Exhibit, Providence, Rhode Island, 16-19 August 2004.

[4] Y. Papadopoulos, J. McDermid, R. Sasse and G. Heiner, "Analysis and synthesis of the behaviour of complex programmable electronic systems in conditions of failure", Reliability Engineering and system safety, Vol. 71, Issue. 3, pp 229-247, March 2001.

[5] Y. Papadopolus, D. Paker and C. Grante, "A method and tool support for model-based semi-automated failure modes and effects analysis of engineering designs", Proceedings of the 9th Australian Workshop on Safety critical systems and software (SCS'04), Vol. 47, pp. 89-95, Australian Computer Society, , Australia, 2004.

[6] D. J. Parker and Y. I. Papadopoulos, "Optimisation of networked control systems using model-based safety analysis techniques", Proceeding of the 2007 IEEE International Conference on Networking, Sensing and Control, pp. 425-430, London, UK, 15-17 April 2007.

[7] A. Lankenau, O. Meyer and B. Krieg-Bruckner, "Safety in robotics: The Bermen autonomous wheelchair", Proceedings of the 5th International workshop on Advanced motion control, AMC '98, Coimbra, pp. 524-529, Portugal, 29 Jun-1 Jul 1998.

[8] J. .K. Kuchar, "Safety analysis methodology for Unmanned Aerial Vehicle (UAV) Collision Avoidance Systems," 6th USA/Europe Seminar on Air Traffic Management Research and Development, Baltimore, MD, June, 2005, (MIT Lincoln Laboratory, Lexington, MA, United States Air Force, #F19628-00-C-0002).

[9] D. M. Guillerez, J. Guiochet and D. Powell, "Experience with model-based user-centered risk assessment for service robots", 12th IEEE international Symposium on High-Assurance Systems Engineering (HASE), pp. 104-113, 2010.

[10] C. J. Claire, I. Mitchell, A. M. Bayen and M. Oishi, "Computational Techniques for the verification of hybrid systems", Proceedings of the IEEE, Vol. 91, No. 7, July 2003.

[11] A. Chutinan and B.H. Krogh, "Verification of infinite-state dynamic systems using approximate quotient transition systems", IEEE Transactions and Automatic Control, Vol. 46, No. 9, pp. 1401-1410, September 2001.

[12] P. Andre and C.M. Edmud, *"Formal verification of curved flight collision avoidance maneuvers: A case study"*, Carnegie Mellon University, Computer Science Department, Paper-1247, 2009.

[13] M. Althoff, O. Stursberg and M.Buss, "Online verification of cognitive car decisions", Proceedings of the IEEE Intelligent Vehicles Symposium, pp. 728-733, Istanbul, Turkey, June 13-15, 2007.

[14] J. Ding, J. H. Gillula, H. Huang, M.P. Vitus, W. Zhang and J.T. Claire, "Hybrid systems in robotics: Toward reachability-based controller design", IEEE Robotics and Automation magazine, September, 2011.

[15] A. A. Julius and G. J. Pappas, "Probabilistic testing for stochastic hybrid systems", Proceedings of the 47th IEEE Conference on Decision and Control, pp. 4030-4035, Cancun, Mexico, Dec. 9-11, 2008.

[16] S. Prajna, A. Jadbabaie and G. J. Pappas, "A framework for worst-case and stochastic safety verification using barrier certificates", IEEE Transactions on Automatic Control, Vol. 52, No. 8, August 2007.

[17] P. Cheng, V. Kumar, "Sampling-based falsification and verification of controllers for continuous dynamic systems", The International Journal of Robotics Research, Vol. 27, No. 11-12, pp. 1232-1245, 2008.

[18] T. Fraichard, "A short paper about motion safety", IEEE International Conference on Robotics and Automation", Roma, Italy, 10-14 April 2007.

[19] H. Choset, K. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. Kavraki, and S. Thrun, *" Principles of robot motion: Theory, Algorithms, and Implementation"*, A Bradford Book, The MIT Press, Cambridge, England, 2005.

[20] S. Karaman, E. Frazzoli, "Sampling-based algorithms for optimal motion planning", The International Journal of Robotics research, 30 (7), pp. 846-894, June 2011.

[21] B.D. Luders, S. Karaman, E. Frazzloi, and J.P. How, "Bounds on tracking error using closed-loop rapidly-exploring random trees", American Control Conference, Baltimore, MD, USA, June 30- July 02 2010.

[22] L. Xiong, F. Xiao-ping, Y. Sheng and Z. Heng, "A novel genetic algorithm for robot path planning in environment containing large numbers of irregular obstacles", ROBOT, Vol. 26, No. 1, Jan 2004.

[23] J. E. Bell and P. R. McMullen, "Ant colony optimization techniques for the vehicle routing problem", Advanced Engineering Informatics 18, pp. 41-48, (2004).

[24] N. Bin, C. Xiong, Z. Liming and X. Wndong, "Recurrent neural network for robot path planning", Proceedings of the 5th Int. Conf. on Parallel and Distributed Computing, Applications and Technologies (PDCAT 2004), LNCS 3320, pp.188-191.

[25] V. K. Banga, R. Kumar and Y. Singh, "Fuzzy-genetic optimal control for robotics systems", Int. J. Physical sciences, Vol. 6(2), pp. 204-212, 18 January 2011

[26] H. Mei, Y. Tian and L. Zu, "A hybrid ant colony optimization algorithm for path planning of robot in dynamic environment", Int. J. of Information Technology, Vol. 12(3), pp. 78-87, 2006.

[27] P. Raja and S. Pugazhenthi, "Path planning for mobile robots to avoid polyhedral and curved obstacles". Int. J. Assistive Robotics and Mechatronics, Vol. 9(2), pp. 31-41, 2008.

[28] A. Sipahioglu, A. Yazici, O. Parlaktuna and U. Gurel, "Real-time tour construction for a mobile robot in a dynamic environment". Robotics and Autonomous Systems, Vol. 56(4), pp. 289-295, April 2008.

[29] P. Raja and S. Pugazhenthi, "Path planning for a mobile robot in dynamic environments", Int. J. of the Physical Sciences, Vol. 6(20), pp. 4721-4731, Sept 2011.

[30] B. Cheol Min, H. Y. Kwon and D. Kim, "Path planning algorithm for VTOL type UAVs based on the methods of ray tracing and limit cycle", Computational Intelligence in Robotics and Automation (CIRA), IEEE International Symposium, pp. 296-301, 15-18 Dec 2009.

[31] S. Griffiths, J. Saunders, A. Curtis and T. McLain, "Obstacle and terrain avoidance for miniature aerial vehicles", IEEE Robotics and Automation Magazine, Vol. 13(3), pp. 34-43, September 2006.

[32] P. O. Pettersson and P. Doherty, "Probabilistic roadmap based path planning for an autonomous unmanned helicopter" Journal of Intelligent and Fuzzy systems, Vol. 17(4), pp. 395-405, September, 2006.

[33] S. A. Bortoff, "Path planning for UAVs". Proceedings of the American Control Conference, Vol. 1, pp. 364-368, June 2000.

[34] T. Lapp and L. Singh, "Model predictive control based trajectory optimization for nap-of-the-earth (NOE) flight including obstacle avoidance", Proceeding of the American Control Conference, Boston, Massachusetts, June 30-July 2 2004.

[35] G. P. Roussos, D. V. Dimarogonas and K. J. Kyriakopoulos, "3D navigation and collision avoidance for a non-holonomic vehicle", American Control Conference, Washington, USA, June 11-13, 2008.

[36] T. Paul, T. R. Krogstad and J. T. Gravdahl, "UAV formation flight using 3D potential field", 16th Mediterranean Conference on Control and Automation, pp. 1240-1245. 25-27 June 2008.

[37] S. C. Han and H. Bang, "Proportional Navigation-based optimal collision avoidance for UAVs". Proceedings of 2nd Int. Con. On Autonomous Robots and Agents, Palmerston North, New Zealand, Dec 13-15 2004.

[38] X. Wang, V. Yadav and S. N. Balakrishanan, "Cooperative UAV formation flying with obstacle/collision avoidance", IEEE Transactions on Control Systems Technology, Vol. 15, No. 4, pp. 672-679, 2007.

[39] J. W. Park, H. D. Oh and M. J. Tahk, "UAV collision avoidance based on geometric approach", SICE Annual Conference, pp. 2122-2126, 2008.

[40] A. Preece, *"Evaluating Verification and Validation Methods in Knowledge Engineering"*, Industrial Knowledge Management, pp. 91-104, Springer, Verlag, 2001.

[41] B. Boehm, "Verifying and validating software requirements and design specifications", IEEE Software, Vol. 1(1), 1984.

[42] T. G. Olson, "Successful verification and validation based on the CMMI^{sm} model", NDIA Systems Engineering Conference, 25 Oct 2005.

[43] M. Debbabi, F. Hassine, O. Jarraya, A. Soeanu and L. Lawneh, "Verification and validation in systems engineering", DOI 10.1007/978-3-642-15228-3, Springer, Verlag, 2010.

[44] R. Arrabales, "Mobile Robots Pioneer", www.conscious-robots.com, April 2007.

[45] O. Balci, *"Verification, Validation and Testing"*, Handbook of simulation, John Wiley & Sons Inc, ISBN 0-471-13403-1, 1998.

[46] T. Eushiuan, "Verification, validation, certification", Carnegie Mellon University, 18-

849b, Dependable Embedded Systems, Spring 1999.
http://www.ece.cmu.edu/~koopman/des_s99/verification/

[47] A. M. Bloch, J. Baillieul, P. Crouch, J. Marsden, Nonholonomic Mechanics and Control, Springer, 2003, ISBN.0-387-95535-6.

[48] R. Siegwart and I. R. Nourbakhsh, "Introduction to Autonomous Mobile Robots", Massachusetts Institute of Technology, 2004.

[49] A.M. Bloch, J. E. Marsdeny, D. V. Zenkovz, Nonholonomic dynamics, Notices of the AMERICAN Mathematical Society, 52 (2005), 324-333.

[50] M. Lepetic, G. Klancar, I. Skrjanc, D. Matko and B. Potocink, "Path planning and path tracking for nonholonomic robots, Mobile Robots", Mobile Robots: New Research, pp. 345-368, ISBN:1-59454-359-3, 2005 Nova Science Publishers.

[51] D. G. Bates, T. Mannchen, R. Kureemun and I. Postlethwaite, "μ-tools for the clearance of flight control laws", American Control Conference, Boston, Massachusetts, June 3- July 2, 2004

[52] L. F. Lee, *"Decentralized motion planning within an artificial potential framework for cooperative payload transport by multi-robot collectives"*, Thesis M.Sc, Dept. of Mechnical and Aerospace Eng, State University of New York, Dec 2004.

[53] A. Mujumdar and R. Padhi, "Nonlinear geometric and differential geometric guidance of UAVs for reactive collision avoidance", Technical report, IISc/SID/AE/LINCGODS/AVARD/2009/01.

[54] S. H. Kim, *"Multi-layer approach to motion planning in obstacle rich environment"*, Master's Thesis, Texas A&M University, 2009, http://hdl.handle.net/1969.1/ETD-TAMU-2621

[55] J. Giesbrecht, "Global path planning for unmanned ground vehicles", Technical memorandum, Defence R&D Canada (DRDC) Suffield, TM 2004-272, Dec 2004.

[56] S. Garrido, L. Moreno, D. Blanco and P. Jurewicz, "Path planning for mobile robot navigation using Voronoi diagram and fast marching", Int. J. of Robotics and Automation (IJRA), Vol. 2(1), 2011.

[57] P. J. McKerrow, Introduction to Robotics, Addision Wesley, 1991.

[58] D. T. Lee and R. L. Drysdale, "Generalization of Voronoi diagram in the plane", SIAM Journal of Computing, Vol. 10(1), pp. 73-83, 1981.

[59] W. A. Kamal, D. W. Gu and I. Postlehwaite *"Safe trajectory planning techniques for autonomous air vehicles"*, Thesis, Leicester University, 2005.

[60] S. Lavalle and J. Kuffner, "Randomized kinodynamic planning", Proceeding of the IEEE Intl. Conference on Robotics and Automation, pp. 473-479, 1999.

[61] Y. Kuwata, J. Teo, G. Fiore, S. Karaman, E. Frazzoli and J.P. How, "Real-time motion planning with applications to autonomous urban driving", IEEE Transactions on Control Systems Technology, Vol. 17(5), pp. 1105-1118, Sept 2009.

[62] D. Polani, B. Browning, A. Bonarini, and K. Yoshida, *"RoboCup 2003: Robot Soccer World Cup VII"*, LNAI 3020, pp.332-343, Springer, Verlag, 2004.

[63] F. Lingelbach, *"Path planning using probabilistic Cell Decomposition*", Licentiate Thesis, Stockholm, Sweden, 2005.

[64] O.Khatib, "Real-time obstacle avoidance for manipulators and mobile robots", Int. Journal Robotics Research, Vol.5(1), pp. 90-99, 1986.

[65] M. C. Joshi, K. M. Moudgalya, "Optimization, Theory and Practice", Alpha Science, 2004.

[66] M. Seda, "Roadmap methods vs. cell decomposition in robot motion planning", Proceedings of the 6th WSEAS International Conference on Signal Processing, Robotics and Automation, Corfu Island, Greece, Feb 16-19 2007.

[67] P. Khosla and R.Volpe, "Superquadratic artificial potentials for obstacle avoidance and approach", IEEE Conference on Robotics and Automations, Philadelphia, April, 1988.

[68] A. Jaklic, A. Leonardis and F. Solina, "Segmentation and recovery of superquadrics", In Computational Imaging and Vision, Vol.20, Kluwer, Dordrecth, 2000.

[69] J. Kim and P. K. Khosla, "Real-time obstacle avoidance using harmonic potential functions", Proceedings of the IEEE Conference on Robotics and Automation, pp. 338-349, June 1992.

[70] E. C. Silva, E. Bicho and W. Erlhagen, "The potential field method and the nonlinear attractor dynamics approach: what are the difference?", 7th Porttuguese Conference on Automatic Control, CONTROLO 2006.

[71] Pioneer 3-DX, http://www.mobilerobots.com/Libraries/Downloads/Pioneer3DX-P3DX-RevA.sflb.ashx

[72] S. S. Ge and Y. J. Cui, "New potential functions for mobile robot path planning", IEEE Transaction on Robotics and Automation, Vol.16(5), pp. 615-620, 2000.

[73] J. Holland, "*Adaption in natural and artificial Systems: An Introductory Analysis with Application to Biology, Control, and Artificial Intelligence"*, University of Michigan press, Ann Arbor, MI, 1975.

[74] S. S. Ge and Y. J. Cui, "Dynamic motion planning for mobile robots using potential field method", Autonomous Robots 13, pp. 207-222, Kluwer Academic Publishers, 2002.

[75] W. E. Kelly, R. Collins, C. Rapids and Iowa, "Conflict detection and altering for separation assurance systems", Digital Avionics Conference, St. Louis, 1999.

[76] J. Krozel and M. Peters, "Strategic conflict detection and resolution for free flight", Proceedings of the 36th IEEE Conference on Decision and Control, Vol. 2, pp. 1822-1828, 10-12 Dec 1997.

[77] J. W. Park, H. D. Oh and M. J. Tahk, "UAV conflict detection and resolution based on geometric approach", Int J.of Aeronautical & Space Science, Vol. 10(1), May 2009.

[78] G. Fasano, D. Accardo and A.Moccia, "Multi-sensor-based fully autonomous non-cooperative collision avoidance system for unmanned air vehicles", Journal of Aerospace computing, Information, and Communication, Vol. 5, October 2008.

[79] C. Carbone, U. Ciniglio, F. Corraro and S.Luongo, "A novel 3D geometric algorithm for aircraft autonomous collision avoidance", Proceeding of the 45thIEEE conference on Decision & Control, USA, December, 13-15, 2006.

[80] E. Masehian and D. Sedighizadeh, "Classic and heuristic approaches in robot motion planning- A chronological review", Proceedings of World Academy of Science, Engineering and Technology. Vol. 23, pp. 101-106, August 2007.

[81] Kalyanmoy Deb, *"Optimization for engineering design: Algorithms and examples"*, Prentice-Hall of India Private Ltd, New Delhi-110 001, 2005.

[82] S. S. Rao, "*Engineering optimization: theory and practice*", 4th edition, John Wiley & Sons, 2009.

[83] G. C. Goodwin, M. M. Seron and J. A. De Dona, *"Constrained control and Estimation- An Optimisation Approach"*, Springer-Verlag, London, 2005.

[84] D. G. Hull, *"Optimal Control Theory for Application"*, Springer-Verlag, New York,

2003.

[85] J. S. Arora, "*Introduction to Optimum Design*", Second Edition, Elsevier Academic Press, 2004.

[86] *fmincon*, http://www.mathworks.co.uk/help/optim/ug/fmincon.html

[87] D. Bates and M. Hagstrom (Eds), *"Nonlinear analysis and synthesis techniques for aircraft control"*, LNCIS 365, pp. 259-300, Springer, 2007.

[88] J. Wang, X. Wu and Z. Xu, "Potential-based obstacle avoidance in Formation Control. Journal of Control Theory and Applications", Vol. 6(3), pp. 311-316, April 2008.

[89] V. Gazi, B. Fidan, Y. S. Hanay and M. I. Koksal, "Aggregation, foraging, and formation control of swarms with nonholonomic agent using potential functions and sliding mode techniques", Turkish Journal of Electrical Engineering and Computer Science. Vol.15(2), pp. 149-168, July 2007.

[90] R. Carona, A. P. Aguiar and J. Gaspar, "Control of unicycle type robots tracking, path following and point stabilization", International Proceeding of the IV Electronics and Telecommunications, pp. 180-185, Lisbon, Portugal, November 2008.

[91] A. D. Luca and G. Oriolo, "Local Incremental Planning for Nonholonomic Mobile Robots", Proceeding of the IEEE International Conference on Robotics and Automation, San Diego, CA, pp. 104-110, 8-13 May 1994.

[92] A. Bemporad, A. D. Luca and G. Oriolo, "Local Incremental Planning for a car-like robot navigating among obstacles", Proceedings of the IEEE International Conference on Robotics and Automation, Minneapolis, MN, pp. 1205-1211, 22-28 April 1996.

[93] J. Pinter, "*Non-convex optimization and its Application-Global Optimization in Action*", Kluwer Academic Publishers, May 1995.

[94] P. Janos,"GlobalOptimization", http://mathworld.wolfram.com/GlobalOptimization.html

[95] J. O. H. Sendín, J. R. Banga and T. Csendes, "Extensions of a Multistart Clustering Algorithm for Constrained Global Optimization Problems", Industrial & Engineering Chemistry Research, Vol. 48(6), pp. 3014-3023, 2009.

[96] D. E. Goldberg, *"Genetic Algorithms in Search, Optimization and Machine Learning",* Addison-Wesley Publishing Co, Readwood City, CA, 1989.

[97] M. Mitchell, "*An introduction to genetic algorithms"*, MIT Press, Cambridge, 1998.

[98] M. Gen and R. Cheng, *"Genetic algorithms and engineering design"*, John Wiley & Sons, 1997.

[99] R. VenkataRao and V. J. Savsani, *"Mechanical design optimization using advanced optimisation techniques"*, Springer Series in Advanced Manufacturing, 2012.

[100] C. G.E. Boender, A. H. G. Rinnooy Kan, G. T. Timmer and L. Stougie, "A stochastic method for global optimization", Mathematical Programming, Vol. 22, pp. 125-140, 1982.

[101] T. Csendes, L. Pal, J. O. H. Sendin and J. R. Banga, "The GLOBAL Optimisation Method Revisited", Optimization Letter, Vol. 2(4), pp. 445-454, 2008.

[102] GLOBAL.m , www.inf.u-szeged.hu/~csendes/regist.php.

[103] R. C. Nelson, *"Flight Stability and Automatic Control"*, Second Edition, 1998.

[104] M. V. Cook, *"Flight Dynamics Principles"*, Second Edition, Elsevier Aerospace Engineering Series, 2007.

[105] Y. C. Paw, *"Synthesis and validation of Flight control for UAV"*, A dissertation, The faculty of the graduate school, The university of Minnesota, December 2009.

[106] C. De La Cruz and R. Carelli, "Dynamic modeling and centralized formation control of mobile robots", Proceedings of the 32nd IEEE Conference on Industrial Electronics, Paris. pp. 3880-3885, November, 2006.

[107] D. R. Jones, "DIRECT Global optimization algorithm", In: Encyclopedi of Optimization, Kluwer Academic Publishers, Dordrecht, Netherlands, pp. 431-440, 2001.

[108] D. E. Finkel and C. T. Kelley, *"Convergence Analysis of the Direct Algorithm"*, Center for Research in Scientific Computation, North Carolina State University, Raleigh, NC, July, 2004.

[109] D. E. Finkel, *"DIRECT optimization algorithm user guide"*, Center for Research in Scientific Computation, North Carolina State University, Raleigh, NC, March, 2003

[110] ISO/CEI, *GUIDE 98-3/SUPP. 1, "Uncertainty of measurement part 3/Supplement 1: Propagations of distributions using a Monte Carlo method"*, ISO/CEI, Switzerland, 2008.

[111] S. Luongo, C. Carbone, F. Corraro and U. Ciniglio, "An optimal 3D analytical solution for collision avoidance between aircraft", *Aerospace conference,* 7-14 March 2009,

doi: 10.1109/AERO.2009.4839595.

[112] J. Kuchar and L. Yang, “A review of conflict detection and resolution modelling methods”, IEEE Transactions on Intelligent Transportation Systems, Vol.1(4), December 2000.

[113] K. D. Bilimoria, “A geometric optimization approach to aircraft conflict resolution”, AIAA Guidance, Navigation, and Control Conference, Denver, Colorado, August 2000.

[114] J. Gross, R. Rajvanshi and K. Subbarao, “Aircraft conflict detection and resolution using mixed geometric and collision cone approaches”, AIAA Guidance, Navigation, and Control Conference and Exhibit, Providence, Rhode Island, 2004.

[115] H. Jie, C. Huaiyan and C. Yun, “Uncertainty evaluation using monte carlo method with Matlab”, The 10th International Conference on Electronic Measurement & Instruments, 2011.

[116] S. Luongo, F. Corraro, U. Ciniglio and V. Di Vito, “A novel 3D analytical algorithm for autonomous collision avoidance considering cylindrical safety bubble”, IEEE Aerospace Conference, MT, USA, 2010.

[117] A. Varga, A. Hansson and G. Puyou (Eds.), *“Optimisation based Clearance of Flight Control Laws”*, Springer, 2012.

[118] P. A. Parrilo, “Software: SOSTOOLS, A sum squares optimization MATLAB toolbox”, http://www.mit.edu/~parrilo/

[119] K. Deb, A. Pratap, S. Agarwal and T. Meyarivan, *“A fast Elitist non-dominated sorting genetic algorithm for multi-objective optimisation, NSGA-11”*, KanGAL report 200001, Indian Institute of Technology, Kanpur, India, 2000.

[120] L. Wang, H. Ishida, T. Hiroyasu and M. Miki, “Examination of Multi-objective optimisation method for global search using DIRECT and GA”, IEEE World Congress on Computational Intelligence, Evolutionary Computation, 2008.