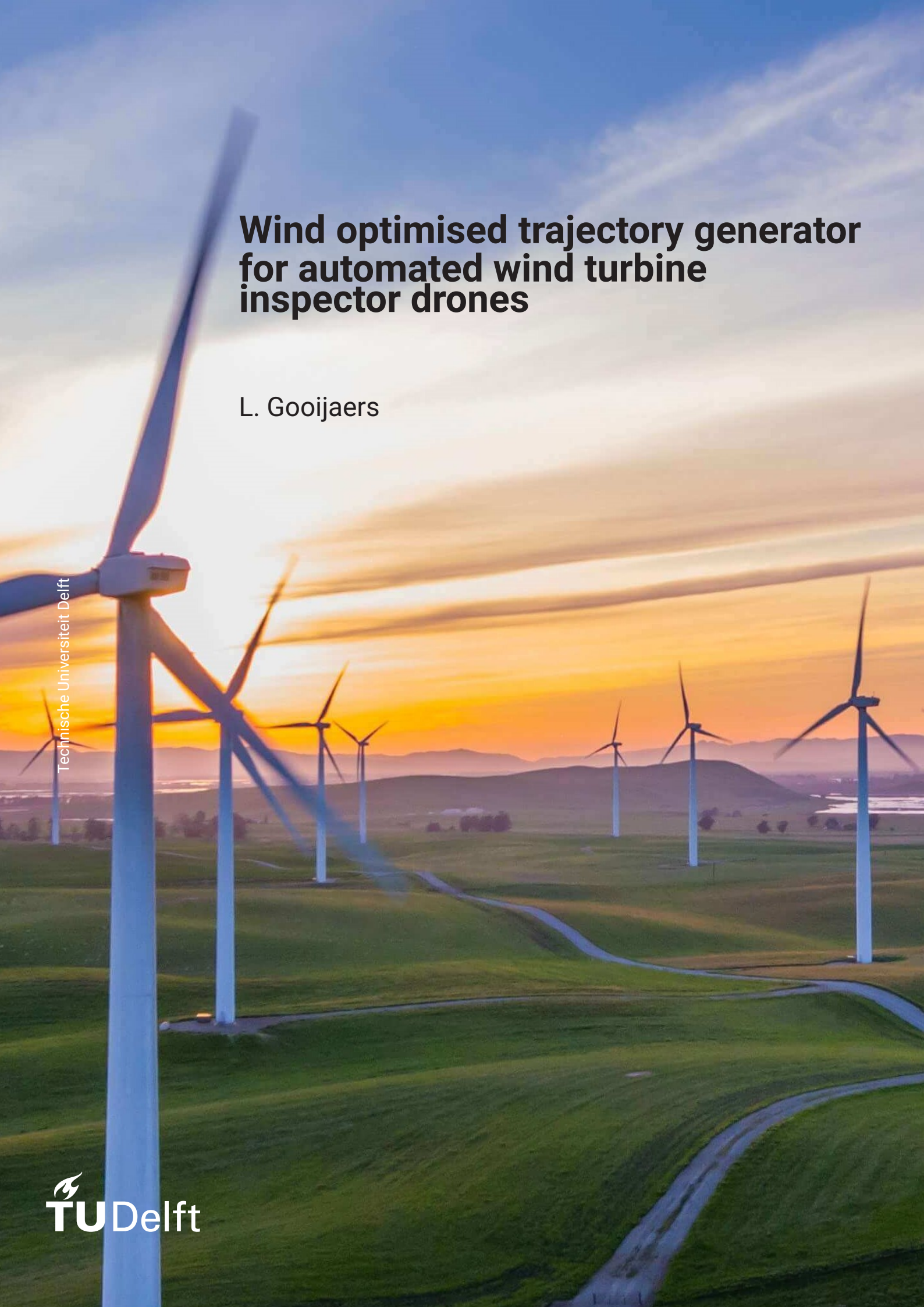


Wind optimised trajectory generator for automated wind turbine inspector drones

L. Gooijaers

Technische Universiteit Delft

TU Delft

Wind optimised trajectory generator for automated wind turbine inspector drones

by

L. Gooijaers

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on wednesday February 17, 2020 at 13:30.

Student number:	4096800	
Project duration:	February 22, 2019 – February 17, 2021	
Thesis committee:	Dr. C. C. de Visser,	TU Delft, supervisor
	Dr. G. C. H. E. de Croon,	TU Delft, chair
	Dr. W. van der Wal,	TU Delft, external member

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft

Preface

This thesis is part of an overarching project which aims to be able to safely and efficiently inspect wind turbines. The goal of this thesis is to map the possibilities and challenges corresponding to trajectory optimisation and provide a possible solution. After having worked on trajectory generation during my internship, this thesis was a logical continuation. The project suited me well due to the concrete problems that had to be solved and direct applicability of the solutions to these problems. During this project I have been able to create a low cost algorithm able to safely handle emergencies, while taking into account the wind field.

While working on my thesis a lot happened on a personal level, both positive and negative, which greatly impacted the duration of the project. I would like to thank my supervisor, Dr.ir. Coen de Visser, for helping me finish this project while allowing me to slow down the pace when necessary. I am also deeply grateful to my wife Mariëlle, who supported my at every time during the project and who was always willing to help wherever possible. Also the positivity and the persistent belief in my abilities of my mother, father an brother were a great support. Lastly I would like to thank my fellow students with who I stayed in touch despite not being able to sit together anymore.

L. Gooijaers
Boekel, January 2021

Contents

1

Introduction

The solutions for automated inspection of wind turbines can be made more efficient while remaining safe and robust. Wind turbines are one of many types of structures which can be inspected using Unmanned Aerial Vehicles (UAV's), resulting in cheaper and faster inspections. Over the last years the application of UAV's has exponentially increased[3][4], resulting in different solutions to the risen challenges of onshore and offshore inspections. Already several companies offer solutions to inspect wind turbines with remotely piloted drones, such as AetosDrones, IAS or ABL Drones. However procedures involving remote pilots are vulnerable to human error and limited by the capabilities of the UAV pilot, which can be solved by performing autonomous inspections as done by for example SkySpecs. Although autonomous inspection is not limited by human capabilities, it still has some challenges. The main challenge is to keep the inspection safe, while remaining time and cost efficient[5].

Most research on autonomous wind turbine inspection focuses on the use of vision as a basis of automated navigation. Most solutions base the navigation on the Hough transform[7] which can be combined with optical flow [3]. Alternatives are performing a LiDaR based mapping prior to inspection[6]. However the most important downside of both the Hough transform and LiDaR based solutions is the increase in operation time due to calibrating or mapping procedures. This thesis is part of a project that uses GPS as basis of navigation to overcome this drawback. A wind turbine position is fixed and the attitude can be obtained from the wind turbine park, hence both can be known prior to the inspection. Using a local GPS transmitter to enhance GPS coverage removes the need for pre-inspection mapping or calibration, decreasing overall operation time and increasing the efficiency of the operation.

The project overarching this thesis has as goal to create a complete, safe and efficient solution for automated wind turbine inspection. This includes all steps from creating a database with the lay-out of available wind turbines to the processing of measurements taken with automated UAV's. This thesis will focus on trajectory generation, which is a key component to ensure that the solution is safe and efficient. The used trajectory generation method is based on the assumption that there is a good knowledge of the environment prior to the inspection. The research objective of this thesis is to create a safe and efficient solution to inspect wind turbines by creating an optimal 3D trajectory generator for automated wind turbine inspector drones. Since implementing an absolute optimal solution is infeasible, the goal is to optimise for travel time and improve in comparison to existing solutions.

To get to this objective several research questions need to be answered. Firstly an overview of the currently used wind turbine inspection methods has to be created. Secondly the state-of-the-art measures to increase UAV safety have to be known. Thirdly the trajectory generation methods suitable for UAV based wind turbine inspection have to be determined. Lastly it has to be determined how both efficiency and safety of wind turbine inspector drones can be increased in comparison to the state-of-the-art methods. The first three questions are answered by and presented in the literature study in chapter 2, while the final question is answered in the article as presented in chapter 3.

Based on the findings in the literature study the first choice of trajectory generation method is numerical optimisation based due to the possibility to create very efficient routes. However this is not feasible since it is a computational expensive solution due to the large trajectory while fast replanning is important for emergency handling. Besides fast replanning it is important that the solution allows for wind optimisation, since the wind field has a large impact on the efficiency of the drone. To be able to accommodate for this

and not become computationally expensive a 2D horizontal path is determined using Dijkstra's algorithm, combined with a separate height profile, while the path tracking is handled by the controller.

The algorithm that is developed to cope with emergency replanning is explained in the article in chapter 3. The contribution of the article is to present a robust trajectory generator for wind turbine inspections which takes into account wind and is able to online create safe emergency routes. The algorithm is tested with three simple scenarios for reliability and efficiency. The used wind cases are simple, but the solution allows for expansion to more difficult optimisation, taking into account effects such as wind shear or wind turbine wakes. The used height optimisation is also simple, however since this is a 2D path planning problem it is possible to use the same or a similar solution as for the horizontal optimisation.

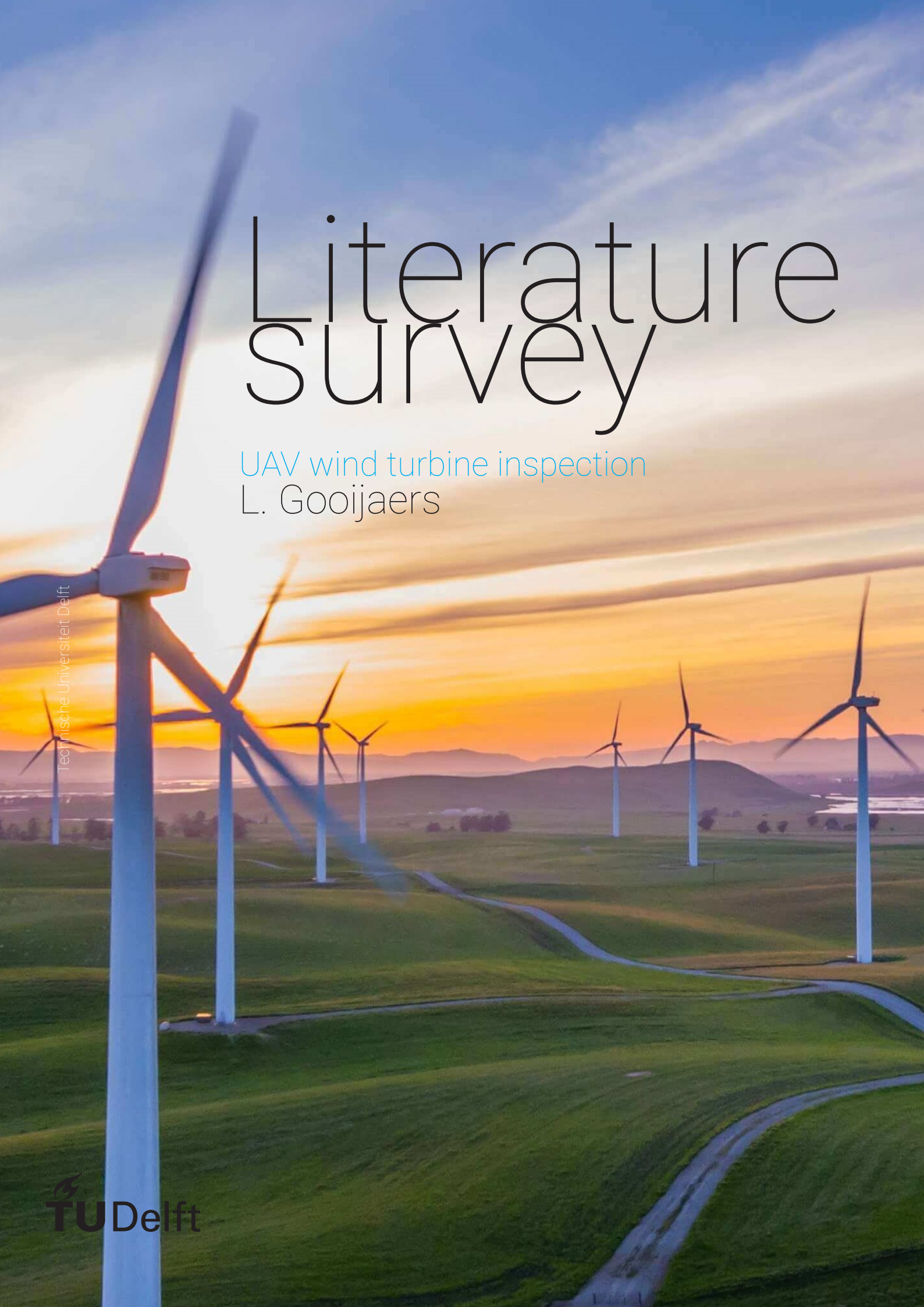
Besides the solution as presented in chapter 3 several additional steps are taken to improve the nominal trajectory generation for the overarching project. The basic non-optimised trajectory planner for nominal operation is optimised by implementing it as a Travelling Salesman Problem. Which can easily be expanded to a Multiple Salesman TSP to allow for multiple drones to increase the efficiency of the operation (total duration of an inspection). Besides this the trajectory generator heavily relies on the controller to follow its generated path. The controller as explained in chapter 3 did not perform optimal for the implementation in wind turbine inspection. To ensure that the drone never crashes due to too large inputs, the velocity and acceleration inputs were limited. Additionally to decrease the chance of overexciting the drone the long distance links of the trajectory are broken up into smaller pieces. This decreases the overall efficiency but allows to developed algorithm to be tested with the provided model and controller.

In this report first the literature study is presented in chapter 2, which includes its own table of contents, conclusion and bibliography. Similarly in chapter 3 an article containing the most important methodology, results and their conclusion is given, accompanied by its own conclusion and bibliography. In chapter 4 the conclusions and recommendations for the entire thesis project are given.

2

Literature survey

In this chapter the literature survey as finished in June 2019 is presented. The survey consists of a general explanation on wind turbine inspection and the approach of the overarching project. This is followed by an explanation of possible methods for trajectory planning. Lastly the research plan is explained, containing a more elaborate explanation of the research objective and research questions.



Literature survey

UAV wind turbine inspection

L. Gooijaers

Technische Universiteit Delft

TU Delft

Contents

1 Introduction

The use of Unmanned Aerial Vehicles (UAV) for inspection of difficult to access structures has increased over the last years. One of the application is the inspection of wind turbines. Already several companies offer solutions to inspect wind turbines, either by remotely piloted[1] or autonomous[2] drones. When remote pilots are used the procedure is vulnerable to human error and limited by the capabilities of the UAV pilot. Autonomous inspection does not have this drawback, however the challenge is to guarantee safe and robust control while staying efficient.

This work is part of a project which has as goal to provide a safe and efficient solution for automated wind turbine inspection. This includes all steps from creating a database with the lay-out of available wind turbines to the processing of measurements taken with automated UAV's. The trajectory generation for the UAVs is a key component to ensure that the solution is safe, robust and efficient, hence this is the main focus of this research. This research aims to create a safe and efficient solution to inspect wind turbines by creating an optimal 3D trajectory generator for automated wind turbine inspector drones.

Due to the many aspects of the project first all aspects of wind turbine inspections are gathered in chapter 2. Based on a short assessment of all the components a general scope is set for further research. The research will focus on trajectory generation, taking into account the wind effects, as explained in chapter 3. Chapter 4 presents the research plan, including research objective, research questions and the methodology. Lastly the conclusion is provided in chapter 5.

[1] AetosDrones, IAS, ABL Drones
[2] SkySpecs

2

Wind turbine inspection

The use of Unmanned Aerial Vehicles (UAV's) for the inspection of structures has exponentially increased over the last years [14]. UAV's are used for inspections in a wide variety of industrial applications [15]. As mentioned in the introduction, one of these applications is the inspection of onshore and offshore wind turbines. To be able to create a safe and efficient solution for automated wind turbine inspection, several aspects have to be looked at. Firstly all aspects of UAV based wind turbine inspection are identified in section 2.1, based on which two main categories can be identified. Based on the first category an overview of the currently used wind turbine inspection methods is given in section 2.2. Based on the second category the available state-of-the-art measures to increase drone safety are explained in section 2.3. Lastly the scope as set for this research is given in section 2.4.

2.1. Project overview

Techniques for non-destructive testing include visual inspection, inspection with a condition monitoring technique and online condition monitoring[3]. An overview of possible condition monitoring techniques is given by Márquez et. al[10] which varies from vibration analysis and acoustic emission to ultrasonic testing or thermography. Since each non-destructive testing technique requires a very different approach, only visual inspection is considered for this research. For visual inspections commonly used approaches are inspections using UAVs, lift, climbing or through telescopic lenses.

Within this project only UAV based approaches are considered. The research to UAV based wind turbine inspection can be divided in two categories, as shown in figure 2.2. The first category contains the parts concerning the UAV itself and the second category contains the parts concerning the inspection. Figure 2.2 also shows the subcategories and components of each subcategory for both.

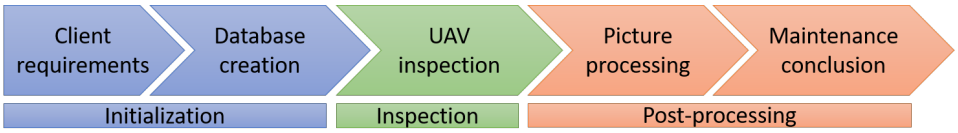


Figure 2.1: Inspection pipeline

For a wind turbine inspection solution several steps have to be taken. A pipeline of these steps is presented in figure 2.1, which shows that each project consist of three main steps. The first step is the initialization of the project, including adapting the approach based in the client requirements and creating a database of the wind turbines for the inspection. After this the inspection can be performed. Lastly the data as gathered during the inspection hase to be processed. If desired it is possible to convert the raw picture data to for example a 3D model or an easy to acces database. Based on the results a maintenance report can be created, which can result in more precise condition monitoring.

2.1.1. Inspection

The first category contains the components that have to do with the inspection, consisting of inspection quality, inspection data and costumer specification. To be able to perform a wind turbine inspection the quality

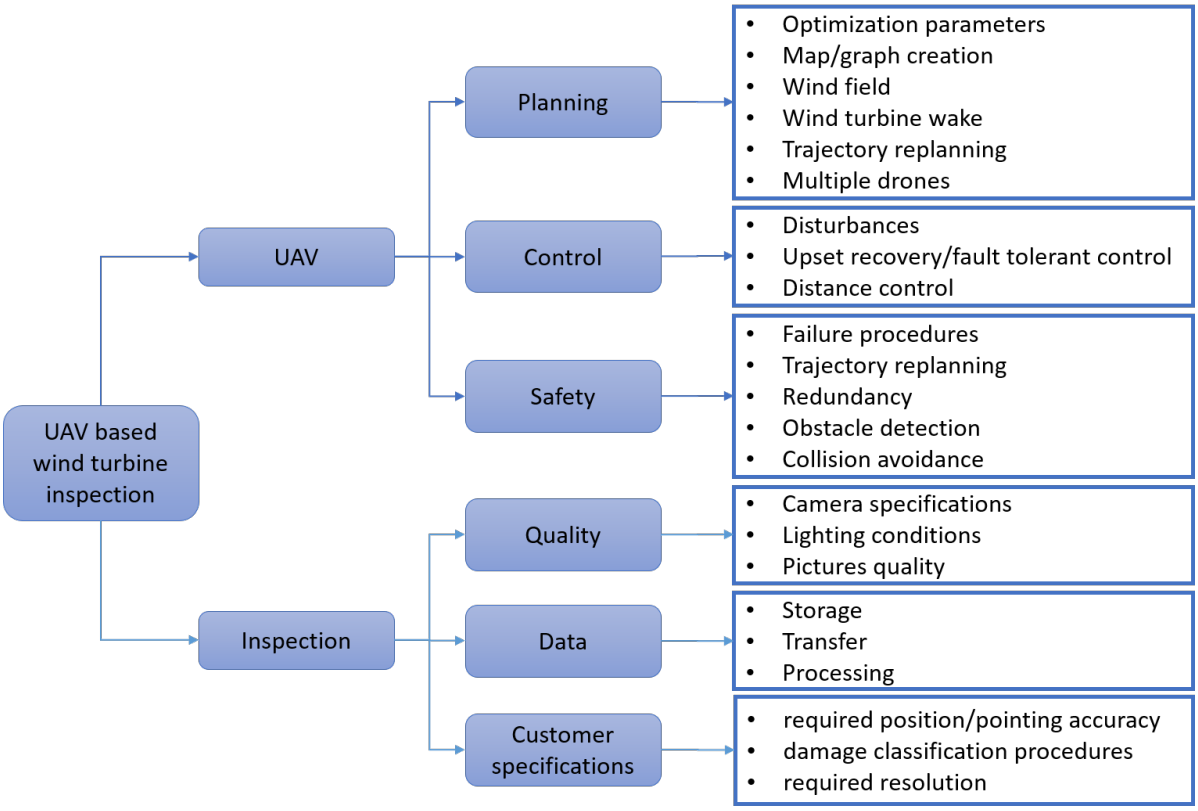


Figure 2.2: Wind turbine inspection overview

of the inspection has to be guaranteed. Since for this project only vision based inspection is considered the quality of the inspection is directly related to the quality of the pictures. The most important factors for this are the camera specifications and lighting conditions.

Next to quality the data handling is part of the inspection routine, consisting of data storage, data transfer and post-processing. Data storage can be either on-board the UAV or at the base station where it can be stored locally or in a cloud service. The data transfer component includes data transfer between all these components. Lastly data processing is an important aspect of the inspection since the pictures as taken by the UAV have to be analysed to asses the wind turbines.

The last subcategory is customer specifications, including required position/pointing accuracy, damage classification procedures and required resolution. These components are closely connected to the available post-processing tools and can vary per customer.

2.1.2. UAV

The second category contains components which are specific for UAV based operations, including UAV control, trajectory planning and safety. The implemented UAV control should be able to cope with external disturbances and keep the UAV at a desired distance from the wind turbine, both for safety and picture quality. The controller should also be able to cope with failure by for example implementing upset recovery procedures or fault tolerant control.

To be able to operate one or more inspection drones its trajectories should be planned. This can be done by optimizing for one of several parameters, such as energy, time or speed. Several aspects can be included during the planning of trajectories such as UAV dynamics, wind and the wind turbine wakes. For the case when something unexpected occurs, such as a shorter battery life or the need to revisit a part of the wind turbine, its trajectory should be replanned. Also the navigation method is important (GPS, vision based, LiDAR). Lastly multiple drones can be used simultaneously to speed up the inspection of wind turbine parks.

When working with UAVs safety is an important aspect and is therefore included as the third subcategory. When the UAV fails, which can be both due to internal and external influences, failure procedures should be implemented taking into account that the wind turbine inspection are done both onshore and offshore.

When due to a failure the calculated trajectories is not sufficient its trajectory should also be replanned. Redundancy of mission critical components can decrease the probability of failure. Also obstacle detection and collision avoidance increase the safety of the system.

2.2. Inspection methods

As mentioned in 2.1, visual inspections can be done by UAVs, lift, climbing or through telescopic lenses. The use of telescopic lenses is decreasing since the UAVs can give a higher accuracy when working on the height of present day wind turbines[25]. Inspection by lift or climbing is expensive in comparison to UAV based inspection and should therefore only be used when a UAV based inspection is insufficient, especially in offshore applications. Lastly robots can be used to inspect the blades, however these techniques are not applied yet. An example of such a robot is given in [8], where a concept is given for an automated inspection system. Since UAVs are much easier and faster to deploy, the same cost disadvantage as for inspection by lift or climbing applies.

When using UAVs to inspect wind turbines two different approaches are used, namely piloted and automated inspection. When using piloted UAVs the procedure is vulnerable to human error and limited by the capabilities of the UAV pilot. Therefore the latest development tend toward autonomous inspection, however the challenge is to guarantee safe and robust control while staying efficient. The only company that currently deploys automated wind turbine inspection drones is SkySpecs[1].

Over the last decade several aspects of UAV based wind turbine inspection has been considered in literature, such as collision avoidance, distance control, vision based automated navigation, tracking of camera reference and waypoint speed constraints. Piers et. al.[19] presents a remote inspection solution for wind turbine blades. The proposed algorithm ensures a set distance to the wind turbine blade is maintained. An on-board high resolution camera combined with a baseline wind turbine model provides excellent results under optimised lighting conditions. However all test were performed indoor and no test were performed under sub-optimal lighting conditions. Hence no information is known on the robustness of the approach and its ability to perform under reduced visibility conditions.

Several studies consider the use of a front facing camera as the sole use of navigation. In [25] the Hough transform is used for detection of the elements of the wind turbine and a Kalman filter was used to track the hub centre. The accuracy and robustness of the approach were proven by experimental results obtained from test at a wind park. However the testing conditions were not stated and it is indicated that the approach is very dependent on the lighting conditions.

In the thesis of S.Høglund [14] a similar approach was used, however it is combined with optical flow and a PID controller. The optical flow is used to avoid collisions with objects, with the Hough transform as a backup when too few tracking features are present. Although a software-in-the-loop simulation was done and all hardware components are tested by implementing some simple tracking tasks, no test are done outside, involving disturbances and non-ideal lighting conditions.

All UAV based inspection approaches do not separate between offshore and onshore, primarily because the wind turbine stays the same except for its size, hence the used algorithms do not change. However there are some differences, namely the wind and the accessibility. Wind speeds offshore are generally higher, but more predictable [1], providing both advantages and disadvantages for UAV control. Increased wind speeds decrease the available flight time (due to increased battery depletion), however more predictable wind speeds give room for better trajectory optimisation. Additionally, due to the increased size of offshore wind turbines, wake effects will be larger, which can decreases flight performance and flight time.

The last important aspect of inspection methods is the quality of the inspection. The companies which currently perform wind turbine inspections do not release any specifications on the quality of their inspections. The desired quality of each inspections can also vary per customer. Despite this some general conclusions can be made. Firstly the customer specifications set the quality of the inspection, which is directly related to the picture quality. Influences on picture quality are movements of the camera or changing light conditions, resulting in motion blur [12]. To minimise unwanted movement cameras can be mounted on a stabilizer, which includes vibration dampers and a fully gimballed system. The effect on changing light conditions can be minimised by the choice of camera.

As mentioned in section 2.1.1 the post-processing of the data also influences the inspection quality. Picture processing can be done by inspecting each picture for damage, but can be made more manageable by first processing the pictures further. This can be done by combining the pictures into a single picture, or into a

[1] https://skyspecs.com *accessed at 20-04-2019*

3D model of the wind turbine such as in [19]. The outcome of the wind turbine assessment can be combined with predictive maintenance to increase cost savings [3].

2.3. UAV control and safety

One of the most important factors for a UAV based inspection method is that it is safe. Increasing UAV safety can be done by decreasing the likelihood or the impact of a failure. The likelihood of a failure can be decreased by implementing a proper controller which can handle external disturbances (such as wind gusts) and can reliably follow a given path (distance control). An example of including wind disturbances is presented by Stepanyan et. al. [24], who presents a unified estimation, navigation and control approach for multi-rotor drones flying in urban environment (hence changing wind fields). An example of a distance control implementation is given by schäfer et. al.[21], who combines a PD distance controller with on-board LiDAR data. Tran et. al[28] shows that besides PID control also a LQR controller is suitable for control in wind fields.

Implementing obstacle detection and collision avoidance is also prerequisite for a safe operation. Depending on the application different sensors are suitable, such as optical or ultrasonic sensors. Ultrasonic sensors have the disadvantage that they do not detect people reliably, while optical sensors fail under poor lighting conditions such as smoke or fog and cannot detect diaphanous obstacles[9]. A combinations of sensors or a restriction on working condition can overcome these problems. Redundancy of these sensors and other safety critical components should be included in the UAV.

Reducing the impact of failure can be done by making sure the UAV stays controllable, which can be done by implementing upset recovery or fault tolerant control. An example of fault tolerant control is the implementation of Adaptive Incremental Nonlinear Dynamic Inversion (INDI) by Smeur et. al [23]. In this paper INDI is implemented on micro air vehicles, which can compensate for a change in control effectiveness (due to failure). Due to a fault tolerant controller mechanical failure can results in a flyable drone with limited capabilities, which allows for trajectory replanning with the new constraints taken into account.

To make sure that the UAV always has a safe exit strategy, failure procedures should be in place. Since such procedures are highly dependent on the UAV, the application and the environment there is no applicable literature available. However the failure procedures should be part of the trajectory generation approach, since replanning is necessary when a mission critical failure occurs. A lot of research is done on trajectory generation and a proper implementation can increase both efficiency and safety. Therefore a more extensive study on this is presented in chapter 3.

2.4. Scope

To limit the scope of this research two fields are looked into more extensively in chapter 3. Firstly trajectory optimisation since this has the most influence on the efficiency of the overall drone performance. Secondly some of the safety measures since when these are properly designed it increases the robustness of the system. For this research an approach will be designed which is a combination of a wind dependent trajectory optimisation and a safety strategy dependent on external influences.

The approach will consist of a 3D trajectory generator which will fit in a larger framework to act as an easy basis for further development. Part of the framework will consist of the trajectory planner in which wind will be taken into account, increasing the overall efficiency of the operation. To improve the safety of the system exit strategies will be implemented coping with a variety of unexpected events such as loss of GPS, structural failure, unexpected high wind gusts and sensor failure. Normal wind gusts or fault tolerant control will not be taken into account since this will be handled by the UAV controller, which is not part of this research. Another option which will not be considered in this research is the use of multiple drones simultaneously. Although this can increase the efficiency significantly, it is not a necessary component to show the functioning of the framework. Figure 2.3 indicates in green the applicable categories and in bold the components that will be taken into account.

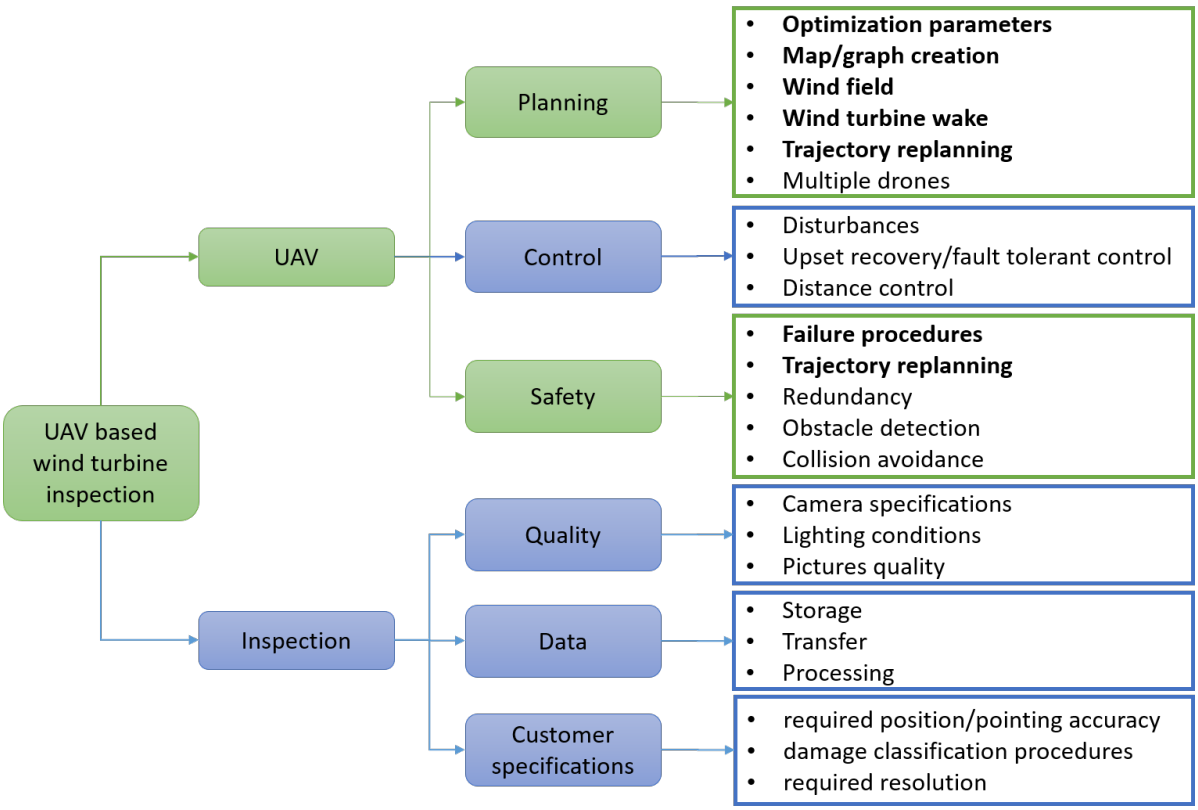


Figure 2.3: Wind turbine inspection overview

3

Trajectory planning

The methods for trajectory planners for quadcopters have increased over the last decade. In section 3.1 the currently available methods are described. To increase the efficiency and safety of trajectory generation the wind field and wind turbine wake are taken into account, which is explained in section 3.2.

3.1. Trajectory generation methods

The simplest method of generating UAV trajectories is by using path planning algorithms, such as graph search (A*[32]) or random search (RRT*). When UAV manoeuvres are at low speeds or the UAV does not need to be at an exact position, but only within a certain range, UAV dynamics can be neglected. However when this is not the case the UAV's dynamical restrictions should be taken into account. The methods that are available for this can be divided in three categories[16], namely boundary value problem (BVP) based, numerical optimisation based and path re-parametrization based. One of the differences is the suitability for online use. When a method is suitable for online use it should be capable to quickly re-plan its path based on a changing environment and do this with limited computational resources. Offline trajectory generation methods are generally too computational expensive to do on-board, however therefore are capable to create more efficient trajectories, taking into account more knowledge such as the wind-field and dynamic constraints.

BVP based methods work by sequentially solving a series of boundary value problems such as presented by Hehn et. al.[13] or by Mueller et. al.[18]. In the paper by Hehn et. al. a trajectory generation algorithm is used that computes high-performance flight trajectories by decoupling the three translational degrees of freedom and ensuring feasibility by deriving decoupled constraints for each degree of freedom. All experiments have been in small spaces, hence aerodynamic effects can be ignored. The decoupling approach decreases the computational time, however also ensures the dynamic capabilities of the UAV cannot be fully used.

Since this aerodynamic effects cannot be ignored when inspecting wind turbines and trajectories can be generated offline it is not an optimal solution. However due to its low computational impact it is a possible solution for situations in which re-planning is necessary and no base station is present. Since the environment is known and a base station is always available offline methods can produce better results for this research. Although re-planning can be done at the base station, with a good computational resources, it still has to be done fast, hence not much longer than about a second to find a feasible solution. BVP methods have another disadvantage since when using other methods, the trajectories can be better optimised for energy consumption, time or UAV properties such as minimum acceleration, snap or jerk.

Numerical optimization methods can be expressed by direct methods, indirect methods or a combination of both[26]. The expressions can then be used to optimise for parameters such as time or total energy consumption. The last decade several numerical optimization approaches have been developed to solve UAV trajectory generation problems[29].

Bry et. al.[5] first uses RRT* to create an optimal path, from which waypoints are selected. After this a piecewise polynomial is optimised that fits through those waypoints, giving a minimum snap trajectory. This paper aims to give a fast solution for aggressive flight in dense indoor environments. Although the environment for wind turbine inspection is very different, these types of algorithms can be useful since they allow for fast replanning.

Rousseau et. al [20] presents a strategy for shooting aerial long takes with a quadrotor, based on minimum jerk trajectory generation. Jerk is chosen to be minimised since jerk quantifies the amplitude of the jolts of a mechanical system, ensuring a good quality of the video. The optimization problem is formulated as a Quadratic Programming (QP) problem and solved by generating a piecewise polynomial trajectory, with variable position, speed and acceleration constraints at the waypoints. Lastly Rousseau uses a receding waypoints horizon methodology to decrease computational load, hence not the exact complete path is calculated, only the next portion of the path. However for safety reasons in this project the complete path should be known, especially in the case when multiple drones are used simultaneously.

Other numerical optimization solution include Lai et. al. [16], which presents a method that uses B-splines to create complex flying trajectories. Mellinger et. al.[17] presents a decoupled minimum snap trajectory generation solution consisting of three steps. First the four decoupled optimization problems are nondimensionalized, second corridor constraints are added end finally the solution is iterated over to optimize for time.

Lastly path re-parametrization methods such as presented by Y.Bouktir et. al.[4] first creates a pure geometric path using primitives such as lines, polynomials or in this case splines. To ensure dynamic feasibility a motion profile is build around this geometric path. In the paper by Y.Bouktir the geometric path is constructed by means of a quintic B-spline model, ensuring continuity of the 2nd order derivatives of sideslip angle and bank angle. Secondly another quintic B-spline model is used to create the motion profile, taking into account velocity constraints.

All methods mentioned above optimize for time, however it can be beneficial to optimize for energy consumption as well. Franco et. al. [7] presents a coverage path planning solution which has constraints on energy and picture resolution. An simple energy model is experimentally obtained for the used drone, resulting in optimal speeds for energy consumption per unit distance. However it might be sufficient to optimize for time while having some speed and acceleration restrictions to avoid excessive battery use.

3.2. Wind field

One of the most important effects on UAV flight time is the wind, hence to increase the efficiency of generated trajectories this should be taken into account. For wind turbine parks two different effects can be considered, namely the overall wind field and the wind turbine wake. Unlike the wind field which can have both a positive and negative effect on flight time, the wind turbine wake will always have a negative effect since it causes turbulence and a decrease in flight time due to the compensating of the controller.

Depending on the type of wind field, constant, linearly varying or unknown, a different approach has to be taken. Silva [22] takes into account constant wind and aerodynamic effects by implementing a three step method. First an optimal waypoint sequence is determined, including constraints due to constant wind. After this the optimal trajectory is planned using quadratic programming and lastly this solution is adapted to fulfil airspeed requirements.

Guerrero et. al[11] presents a path planning approach for both constant and linearly varying wind fields. When the wind field is constant and there are no restrictions on the UAVs start and final heading the time optimal trajectory consist of straight lines. When the wind field is linearly varying the trajectory can be determined by simple formulas.

However for a case of uniform winds where the start and final heading of the UAV is important and a maximum turn rate is taken into account Techy et. al. [27] shows that the trajectory consist of straight lines and trochoidal path segments. Techy et. al investigates the switching points between the straight lines and trochoidal path segments resulting in a time optimized solution.

When the wind field is unknown it cannot be taken into account during trajectory generation. However Xiang et. al.[31] presents a derivation that determines the wind speed and direction based on a quadrotor dynamics model. This can aid in determining the wind field at places that have no wind sensors or are difficult to reach, however it is not ideal since the measurements are noisy. For better results the UAV should hover, which undesirably increases flight time.

Since wind turbine parks have a relative constant horizontal wind field, especially offshore, it is expected that it is not necessary to use a UAV to identify the wind field. Current wind data can be read from wind sensors on the wind turbines and/or at the ground. A simple model can be used to estimate the three dimensional wind field by taking into account the wind gradient. An approach such as from Guerrero et. al.[11] can probably give very good results if combined with dynamically feasible trajectories such as in Silva et. al.[22]. Depending on the type of trajectories that will be used during inspection the addition of trochoidal

path segments such as by Techy et. al.[27] might not be necessary.

Wind turbine wake can also be taken into account when planning trajectories through a wind park. This is important in the case not all wind turbines are inspected hence some can be turned on. Several papers have modelled the effects of wind turbine wakes such as Vermeer et. al.[30] or Bastankhah et. al.[2]. However advanced models of the wind turbine wake are probably not needed, however knowing the shape of the wakes gives the option to avoid them if possible by flying over or under the wake. When inspections happen to be in the wake of another wind turbine, a reduced flight period can be expected due to the increased turbulence compensation that is needed. It can also be taken into account when determining the safe minimum distance between the UAV and the wind turbine.

3.3. UAV trajectories for wind turbine inspection

The challenge in generating trajectories for wind turbine inspection is the combination of the wind turbine inspection itself and the manoeuvring between the wind turbines, since both have different requirements on the trajectories and UAV. The combined trajectory has to be determined fast enough to safely be able to replan a route when desired. Due to the large distances between wind turbines and absence of obstacles other than the wind turbines the trajectories in between wind turbines can be relatively simple and smooth. Therefore the UAV dynamics might not have to be taken into account directly, smooth continuous trajectories might be sufficient. Around the wind turbines the UAV manoeuvres have to be more precise, hence UAV dynamics probably have to be taken into account during planning. It should be possible to (re)plan a trajectory within approximately one second. To be able to do this the trajectory generation can be split up in the following three parts like in the paper by Silva et. al.[22] (section 3.2.

The first step is finding the optimal sequence of waypoints, which is predetermined for the wind turbine inspections itself, but can be optimized for flying in between wind turbines. The latter can be seen as a Travelling Salesman Problem of which the dimension (number of wind turbines to visit) remains small due to the limited number of wind turbines. The outcome is the order in which the wind turbines have to be visited.

The second step is determining the trajectory between these waypoints, in which again a distinction is made between inspection and flying between wind turbines. For the inspection an approach is needed that allows for corridor constraints to ensure the quality and safety of the inspection. There is limited possibility to optimize for time or energy due to the speed and corridor restrictions during the inspection, however the UAV should be able to follow the trajectory accurately. An approach which allows for this is presented by Mellinger et. al.[17] and shown in section 3.3.2. However if this approach is to computationally expensive to allow for quickly replanning a more simple solution could suffice, such as only determine polynomial function to move around the hub and tip and use straight segments otherwise.

For flight in between wind turbines the UAV covers relatively large distances in an environment with (almost) no obstacles besides the wind turbines. Combined with a relatively low number of waypoints it allows for methods which optimize for energy or speed. A method which is suitable for both and allows for replanning is presented by Chamseddine et. al. [6]. Just as the method presented by Mellinger et. al. it uses the flateness property of a quadrotor. The used quadrotor model including the flatness property is explained in section 3.3.1 and is the same as used by Mellinger et. al..

The third and last step is to ensure that the maximum UAV velocity is never exceeded. When using a method as presented by Mellinger et. al. this can be done by incrementally giving or taking time to segments without having to solve the quadratic programming problem again. This step is not necessary when using the method as presented by Chamseddine et. al. [6].

However as mentioned before when flying in between wind turbines, the trajectories are long and do not require aggressive manoeuvring. To increase the robustness of the solution a simpler approach than presented by Chamseddine et. al. can be used for this. To have energy optimized trajectories the wind field and the UAV speed are the most important factors. Due to the relatively long distances the UAV can be flown at a constant speed throughout the trajectory. The ideal speed (least energy consumption) can be found similar as been done by Di Franco et. al.[7]. For the wind field a relatively simple model can be used, which decouples the horizontal and vertical navigation. For the horizontal plane a constant wind can be assumed, with penalties for flying through a wind turbine wake. As proven by Techy et. al. [27] a optimal route through a constant wind field consist of straight lines. For vertical navigation an additional factor can be taken into account since at different heights the wind will not be constant due to vertical wind shear. This can be measured or has to be modelled, depending on the available wind sensors. An often used model for the change in wind for wind turbine design is the following.

$$v(h) = v_{10} \cdot \left(\frac{h}{h_{10}}\right)^a \tag{3.1}$$

Where v_{10} and h_{10} are respectively the velocity and height at 10 meters altitude. The Hellmann exponent (a) indicates the amount of shear and varies from 0.06 for unstable air above open water to 0.60 for stable air above cities.

3.3.1. Quadrotor model

A commonly used simplified quadrotor model[6] uses a body fixed frame attached to the center of gravity of the quadrotor, with the z-axis pointing upwards. The second reference frame is the intertial frame, to which the body fixed reference frame is related with a position vector [x,y,z] and the Euler angles (ϕ, θ, ψ). The rotation matrix R is given by equation 3.2, where c and s are short for cos and sin.

$$R = \begin{bmatrix} c\phi c\theta & c\phi s\theta s\psi - s\phi c\psi & c\phi s\theta c\psi + s\phi s\psi \\ s\phi c\theta & s\phi s\theta s\psi + c\phi c\psi & s\phi s\theta c\psi - c\phi s\psi \\ -s\theta & c\theta s\psi & c\theta c\psi \end{bmatrix} \tag{3.2}$$

The six output states $[x, y, z, \theta, \psi, \phi]$ can be written as a function of the four input forces produced by the propellers are P_1 to P_4. The resulting equations of motion can be written as follows.

$$\begin{aligned} \ddot{x} &= \frac{(P_1 + P_2 + P_3 + P_4)(c\phi s\theta c\psi + s\phi s\psi) - K_1 \dot{x}}{m} \\ \ddot{y} &= \frac{(P_1 + P_2 + P_3 + P_4)(s\phi s\theta c\psi - c\phi s\psi) - K_1 \dot{y}}{m} \\ \ddot{z} &= \frac{(P_1 + P_2 + P_3 + P_4)(c\phi c\psi) - K_3 \dot{z}}{m} \\ \ddot{\theta} &= \frac{l(-P_1 - P_2 + P_3 + P_4 - K_4 \dot{\theta}}{J_1} \\ \ddot{\psi} &= \frac{l(-P_1 + P_2 + P_3 - P_4 - K_5 \dot{\psi}}{J_2} \\ \ddot{\phi} &= \frac{(M_1 - M_2 + M_3 - M_4 - K_6 \dot{\phi}}{J_3} \end{aligned} \tag{3.3}$$

Where K_i are the drag coefficients, which can be assumed to be zero for low speeds. J_i are the moments of inertia and m is the quadrotor mass. For simplicity, the inputs can be written as follows.

$$\begin{aligned} u_1 &= \frac{(P_1 + P_2 + P_3 + P_4)}{m} \\ u_2 &= \frac{l(-P_1 - P_2 + P_3 + P_4)}{J_1} \\ u_3 &= \frac{l(-P_1 + P_2 + P_3 - P_4)}{J_2} \\ u_4 &= \frac{((M_1 - M_2 + M_3 - M_4)}{J_3} \end{aligned} \tag{3.4}$$

Resulting in the following equations of motion.

$$\begin{aligned} \ddot{x} &= u_1(c\phi s\theta c\psi + s\phi s\psi) \\ \ddot{y} &= u_1(s\phi s\theta c\psi - c\phi s\psi) \\ \ddot{z} &= u_1(c\phi c\psi) \\ \ddot{\theta} &= u_2, \quad \ddot{\psi} = u_3, \quad \ddot{\phi} = u_4 \end{aligned} \tag{3.5}$$

To simplify the model further hovering condition can be assumed, hence $u_1 \approx g$. Combined with small angles θ and ϕ and a constant yaw angle, the equations 3.5 that results in the following flat system states.

$$\begin{aligned} \ddot{x} &= g\theta, \quad \ddot{y} = -g\phi, \quad \ddot{z} = -g + u_1 \\ \ddot{\theta} &= u_2, \quad \ddot{\psi} = u_3, \quad \ddot{\phi} = u_4 \end{aligned} \tag{3.6}$$

Differential flatness can be used to simplify the trajectory generation problem. Using the flatness of a quadrotor allows to conclude whether or not a computed trajectory is feasible. Commonly used flat outputs for a quadrotor are $F_1 = z$, $F_2 = x$, $F_3 = y$ and $F_4 = \psi$. From equation 3.6 the other states can be found to be the following.

$$\theta = \frac{\ddot{F}_2}{g}, \quad \psi = -\frac{\ddot{F}_3}{g} \tag{3.7}$$

The control inputs can also be expressed in terms of the flat outputs based on equation 3.6.

$$u_1 = F_1 \ddot{+} g, \quad u_2 = \frac{F_2^{(4)}}{g}, \quad u_3 = -\frac{F_3^{(4)}}{g}, \quad u_4 = \ddot{F}_4 \tag{3.8}$$

3.3.2. Optimal trajectory optimization

To come up with an optimized trajectory Mellinger et. al. [17] bases its method on keyframes, which are positions in space combined with a yaw angle. In between keyframes there is a safe corridor to navigate through. The goal is to create a smooth trajectory, which is defined as:

$$\sigma(t) : [t_0, t_m] \rightarrow \mathbb{R}^3 \times SO(2) \tag{3.9}$$

To come up with smooth trajectories, they are written as piecewise polynomial functions of order n over m time intervals, resulting in the following equation.

$$\sigma_T(t) = \begin{cases} \sum_{i=0}^{n} \sigma_{Ti1} t^i & t_0 \leq t < t_1 \\ \sum_{i=0}^{n} \sigma_{Ti2} t^i & t_1 \leq t < t_2 \\ \vdots & \\ \sum_{i=0}^{n} \sigma_{Tim} t^i & t_{m-1} \leq t < t_m \end{cases} \tag{3.10}$$

To solve this problem an optimization program is made which minimizes the integral of the k_rth derivative of position squared and K_ϕth derivative of yaw angle squared, with $\mathbf{r} = [x, y, z]^T$. The program without corridor constraints is the following:

$$\begin{aligned} min \int_{t_0}^{t_m} \mu_r \left\| \frac{d^{k_r} \mathbf{r_T}}{dt^{k_r}} \right\|^2 + m u_\psi \frac{d^{k_\psi} \psi_T}{dt^{k_\psi}}^2 dt \\ \text{subject to:} \\ \sigma_T(t_i) = \sigma_i, \quad i = 0, ..., m \\ \left. \frac{d^p x_T}{dt^p} \right|_{t=t_j} = 0 \text{ or free}, \quad j = 0, m; p = 1, ..., k_r \\ \left. \frac{d^p y_T}{dt^p} \right|_{t=t_j} = 0 \text{ or free}, \quad j = 0, m; p = 1, ..., k_r \\ \left. \frac{d^p z_T}{dt^p} \right|_{t=t_j} = 0 \text{ or free}, \quad j = 0, m; p = 1, ..., k_r \\ \left. \frac{d^p \psi_T}{dt^p} \right|_{t=t_j} = 0 \text{ or free}, \quad j = 0, m; p = 1, ..., k_\psi \end{aligned} \tag{3.11}$$

In this equation $\sigma_T = [x_T, y_T, z_T, \psi_T]^T$, $\sigma_i = [x_i, y_i, z_i, \psi_i]^T$ and both μ_r and μ_ψ make the integrand nondimensional. The problem is written as a quadratic program by writing $\sigma_{T_{ij}} = [x_{T_{ij}}, y_{T_{ij}}, z_{T_{ij}}, \psi_{T_{ij}}]^T$ as a $4nm \times 1$ vector c with decision variables $\{x_{T_{ij}}, y_{T_{ij}}, z_{T_{ij}}, \psi_{T_{ij}}\}$:

$$\begin{aligned} \min \quad & c^T H c + f^T c \\ \text{subject to} \quad & Ac \leq b \end{aligned} \tag{3.12}$$

The objective function makes sure that snap is minimized while the constraint can incorporate the constraints on states and inputs. Any condition (Initial, final or intermediate) can be written as an inequality constraint. As mentioned before corridor constraints can be added. To do this the perpendicular distance vector from segment i is defined as follows.

$$d_i(t) - (\mathbf{r}_T(t) - \mathbf{r}_i) - ((\mathbf{r}_T(t) - \mathbf{r}_i) \cdot \mathbf{t}_i)\mathbf{t}_i \tag{3.13}$$

In this equation $\mathbf{t}_i$ is defined as the unit vector along a segment from $\mathbf{r}_i$ to $\mathbf{r}_{i+1}$. The constraint can then be added as follows, with equivalents for $\mathbf{Y}_W$ and $\mathbf{Z}_W$ (the non-denationalised position).

$$\left| \mathbf{X}_W \cdot \mathbf{d}_i \left(t_i + \frac{j}{1+n_c}(t_{i+1} - t_i) \right) \right| \leq \delta_i \text{ for } j = 1, ..., n_c \tag{3.14}$$

Where δ_i is the corridor width on the infinity norm. Mellinger et. al present an additional step to make the trajectories time optimizaed, which might not be necessary for wind turbine inspection. The time optimization is a second minimization problem presented in equation 3.15. Here T_i is the time it takes to complete segment i.

$$\begin{aligned} &\min f(\mathbf{T}) \\ &\text{subject to} \quad \sum T_i = t_m \\ &\qquad\qquad\ \ T_i \geq 0 \end{aligned} \tag{3.15}$$

4

Research plan

Based on the literature survey the research objectives and questions can be formed. Both are presented in 4.1, followed by the resulting methodology in section 4.2.

4.1. Research objective and questions

This research is part of a project of which the goal is to create a new wind turbine inspection method. This research focuses on the trajectory generation of this project. Therefore the research objective is formulated as follows:

Create a safe and efficient solution to inspect wind turbines by creating an optimal 3D trajectory generator for automated wind turbine inspector drones.

An important note to this objective is that a 3D trajectory can be optimal in different ways. In this research both time and energy are investigated and a trade-off will be made which is most important for the application. This also is dependent on the wishes of a client. Besides this a truly optimal solution is computationally very expensive hence undesirable. Ideal is when a near optimal solution can be found within a matter of seconds, hence this will be the goal. To be able to reach this research objective several research questions need to be answered. These questions are designed around some key components of this objective, namely the inspections, UAV safety and efficient trajectory optimization. Questions 1, 2 and 3 are answered based on the literature survey as done in chapters 2 and 3 and the conclusion on this is given in chapter 5.

1. What methods are currently used for UAV based wind turbine inspections?
   (a) What wind turbine inspection methods are available?
   (b) What is the difference between offshore and onshore inspection?
   (c) How is the quality of the inspection guaranteed?
2. What are state-of-the-art measures to increase UAV safety?
   (a) How can a UAV be controlled in the presence of external disturbance?
   (b) How can a UAV be safely controlled when a failure occurs?
3. Which trajectory generation methods are suitable for UAV wind turbine inspection?
   (a) What is the state-of-the-art of trajectory generation methods for drones?
   (b) What is the effect of including UAV dynamics during trajectory generation?
   (c) Which methods are suitable for mission or safety related replanning?
   (d) How can wind turbine position survey errors be taken into account?
4. How can both efficiency and safety of wind turbine inspector drones be increased in comparison to the state-of-the-art methods?
   (a) How can the efficiency of the inspection of an entire wind farm be improved regarding trajectory generation?
   (b) How can the presence of wind be included to increase safety and efficiency?

(c) How can be coped with drone failure during inspection?
(d) What is the role and influence of the camera system?

4.2. Methodology

Since this research is part of a project the first step is to create a general interface which will fit any future work as well. The 3D trajectory generator will be a part of a larger framework and its interface should accommodate any future work. As mentioned in section 4.1 separate approaches will be used for optimizing the trajectories during inspection of a wind turbine and during flying in between the wind turbines. The framework will also be able to provide the option to fly with multiple drones, although this will not be implemented in this research. The trajectory generator will take into account the following aspects:

- Speeds constraints on waypoints
- Trajectory replanning in case of mission change
- Trajectory replanning in case of failure (exit strategies)
- Wind field
- Incorrect survey data
- Incorrect external wind turbine data (hub and blade position)

A flow diagram of the components used during inspection is shown in figure 4.1. The trajectory generator will be part of the base station, which will communicate to the client input (wind turbine data), available wind sensors and the UAV(s). The trajectories as generated at the base station are send to the UAV, which will perform the inspection. During the inspection the UAV will send position, attitude and status data to the base station, based on which the station can replan any trajectories if needed. The UAV control will be automated and based on the given trajectory and onboard sensors.

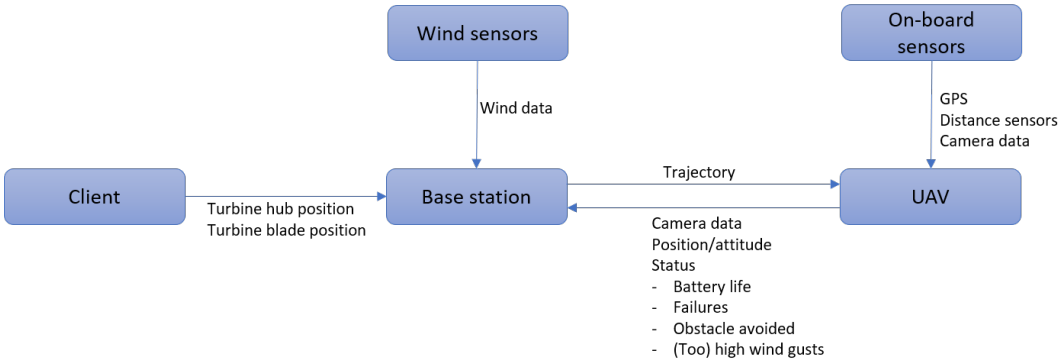


Figure 4.1: Data flow during inspection

To make sure the framework works at least the following tests need to be performed:

1. Replanning speed
2. Trajectory optimization indoor on a small UAV
3. Trajectory optimization outdoor, both with and without (hard) wind present
4. Trajectory optimization on the UAV which will be used during inspection
5. Usability of GPS as sole purpose of navigation around wind turbines
6. Laser Range Finder accuracy
7. Picture quality

Test 1 has to be done to verify that in every situation the implementation is fast enough to provide the UAV with a save trajectory. Tests 2 to 4 need to be done to verify that the algorithm works as desired. The goal is to be able to navigate in GPS, however this is not yet done around wind turbines, hence a test (5) needs to be done to check that it is feasible. Test 6 is done since for the creation of the wind turbine database a laser range finder will be used, which is also new for wind turbine inspection. Lastly to validate that the resulting implementation is as desired, the picture quality has to be tested.

Depending on the availability of the inspection UAV, equipment and wind turbines tests 4 to 7 can be done after this research. These are not needed to finish this research but are needed to finish the overall project.

5

Conclusion

This literature survey is done to create a basis to be able to develop a safe and efficient solution to inspect wind turbines by creating an optimal 3D trajectory generator for automated wind turbine inspector drones. The first three research questions as given in section 4.1 are answered by this report.

The currently available wind turbine inspection methods (question one) are by UAVs, lift, climbing or telescopic lenses. The latter is decreasing, while UAV's are much more common, both piloted and automated. For some inspections lifts or climbing will always be necessary, but for visual inspection UAV are the most efficient option. For inspection there are a few difference between onshore and offshore. Offshore winds are stronger but more constant and offshore accessibility can be restricted. Besides this the wind turbines are generally larger, however the same algorithms can be used for automated inspection. Lastly for question one, the quality of the inspection is largely dependent on camera quality and post-processing of the data, both which are outside the scope of this project.

When considering UAV safety (question two) the presence of external disturbances such as wind are important. Wind gusts are dealt with by the on-board controller, which is outside the scope of this project. However since a controller will not be able to counter all the wind gusts, a minimum safe distance should be taken into account when generating trajectories. Also failure should be taken into account, which can be done by implementing fault tolerant control and upset recovery, combined with trajectory replanning. Within this project failure will be taken into account by creating exit strategies based on reduced UAV capabilities.

State-of-the-art of trajectory generation methods (question three) can be a simple path planning algorithm or a more extensive trajectory optimization methods. Trajectory optimization methods can be divided in BVP methods, path re-parametrization methods and numerical optimization methods. Of these methods the latter forms the trend of the last years and gives the most optimal results, however it is also the most computationally expensive. Including UAV dynamics and wind increases the usability of the solution, making it possible to better optimize for time, energy or UAV properties such as jerk or snap. For replanning numerical optimization methods can be made suitable if the aproach is split up in smaller problems.

For this project UAV wind turbine inspection can be separated in two types of trajectories. Firstly flying around in the wind park, hence between start/end position and wind turbines, which covers large distances and has no impact on the quality of the inspection. Secondly the inspection itself, where the UAV flies close to the wind turbine. Both types of navigation can benefit from a different type of trajectory generation method. When large distance are covered and no dense grit of waypoints has to be followed it is important to take into account the wind field. For this type of flight a simple strategy can be used, which takes into account the wind field and the UAV speed, hence optimizing for energy consumption per unit distance.

Also the basis for navigation should be considered, for which the trend is to use vision based navigation and a single time LiDAR. However a more accurate and robust approach can be to use GPS. Since all inspections are outdoors and the inspection are not in cluttered environment, it is expected that the available GPS signal is sufficient for navigation, hence providing a fast and universal solution.

Lastly the solution which will be chosen has to be able to cope with measurement errors. The sizing and positioning of the wind turbines will be predetermined using a method like the laser range finder. To make sure measurement errors have little impact on the safety of the system additional sensors has to be used to confirm the database. This can be done by distance measurement sensors or vision based sensors and including a safety margin.

Bibliography

[1] Spenser Anderson. Comparing Offshore and Onshore Wind. Technical report, The Economics of Oil and Energy, 2013. URL http://pages.hmc.edu/evans/andersonwind.pdf.

[2] Majid Bastankhah and Fernando Porté-Agel. A new analytical model for wind-turbine wakes. *Renewable Energy*, 70:116–123, 10 2014. ISSN 09601481. doi: 10.1016/j.renene.2014.01.002. URL https://linkinghub.elsevier.com/retrieve/pii/S0960148114000317.

[3] François Besnard and Lina Bertling. An Approach for Condition-Based Maintenance Optimization Applied to Wind Turbine Blades. *IEEE TRANSACTIONS ON SUSTAINABLE ENERGY*, 1(2):77, 2010. doi: 10.1109/TSTE.2010.2049452. URL https://www.researchgate.net/publication/236875104.

[4] Y. Bouktir, M. Haddad, and T. Chettibi. Trajectory planning for a quadrotor helicopter. In *2008 Mediterranean Conference on Control and Automation - Conference Proceedings, MED'08*, pages 1258–1263. IEEE, 6 2008. ISBN 9781424425051. doi: 10.1109/MED.2008.4602025. URL http://ieeexplore.ieee.org/document/4602025/.

[5] Adam Bry, Charles Richter, Abraham Bachrach, and Nicholas Roy. Aggressive flight of fixed-wing and quadrotor aircraft in dense indoor environments. *The International Journal of Robotics Research*, 34(7):969–1002, 2015. doi: 10.1177/0278364914558129. URL https://journals.sagepub.com/doi/pdf/10.1177/0278364914558129.

[6] Abbas Chamseddine, Youmin Zhang, Camille Alain Rabbath, Cedric Join, and Didier Theilliol. Flatness-based trajectory planning/replanning for a quadrotor unmanned aerial vehicle. *IEEE Transactions on Aerospace and Electronic Systems*, 48(4):2832–2847, 2012. ISSN 00189251. doi: 10.1109/TAES.2012.6324664.

[7] Carmelo Di Franco, Giorgio Buttazzo, C Di Franco, and G Buttazzo. Coverage Path Planning for UAVs Photogrammetry with Energy and Resolution Constraints. *Journal of Intelligent & Robotic Systems*, 83:445–462, 2016. doi: 10.1007/s10846-016-0348-x. URL https://link.springer.com/content/pdf/10.1007%2Fs10846-016-0348-x.pdf.

[8] Norbert Elkmann, Torsten Felsch, and Tilo Förster. Robot for rotor blade inspection. *2010 1st International Conference on Applied Robotics for the Power Industry, CARPI 2010*, pages 1–5, 2010. doi: 10.1109/CARPI.2010.5624444.

[9] Nils Gageik, Paul Benz, and Sergio Montenegro. Obstacle detection and collision avoidance for a UAV with complementary low-cost sensors. *IEEE Access*, 3:599–609, 2015. ISSN 21693536. doi: 10.1109/ACCESS.2015.2432455.

[10] Fausto Pedro García Márquez, Andrew Mark Tobias, Jesús María Pinar Pérez, and Mayorkinos Papaelias. Condition monitoring of wind turbines: Techniques and methods. *Renewable Energy journal*, 2012. doi: 10.1016/j.renene.2012.03.003. URL https://www.sciencedirect.com/science/article/pii/S0960148112001899.

[11] J. A. Guerrero, J. A. Escareno, and Y. Bestaoui. Quad-rotor MAV trajectory planning in wind fields. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 778–783, 2013. ISSN 10504729. doi: 10.1109/ICRA.2013.6630661.

[12] Norman Hallermann and Guido Morgenthal. Visual inspection strategies for large bridges using Unmanned Aerial Vehicles (UAV). In *7th International Conference on Bridge Maintenance, Safety and Management*. IABMAS, 2014. doi: 10.1201/b17063-96. URL https://www.researchgate.net/publication/269671543.

[13] Markus Hehn and Raffaello Dandrea. Real-Time Trajectory Generation for Quadrocopters. *IEEE Transactions on Robotics*, 31(4):877–892, 2015. ISSN 15523098. doi: 10.1109/TRO.2015.2432611.

[14] Sondre Hoglund. *Autonomous Inspection of Wind Turbines and Buildings using an UAV*. PhD thesis, Norwegian University of Science and Technology - Trondheim, 2014. URL http://hdl.handle.net/11250/261287.

[15] Sophie Jordan, Julian Moore, Sierra Hovet, John Box, Jason Perry, Kevin Kirsche, Dexter Lewis, Zion Tsz, and Ho Tse. State-of-the-art technologies for UAV inspections. *IET Radar, Sonar & Navigation*, 2017. ISSN 1751-8784. doi: 10.1049/iet-rsn.2017.0251. URL www.ietdl.org.

[16] Shupeng Lai, Kangli Wang, and Ben M. Chen. Dynamically feasible trajectory generation method for quadrotor unmanned vehicles with state constraints. *Chinese Control Conference, CCC*, pages 6252–6257, 2017. ISSN 21612927. doi: 10.23919/ChiCC.2017.8028351.

[17] Daniel Mellinger and Vijay Kumar. Minimum snap trajectory generation and control for quadrotors. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 2520–2525, 2011. ISSN 10504729. doi: 10.1109/ICRA.2011.5980409.

[18] Mark W. Mueller, Markus Hehn, and Raffaello D'Andrea. A computationally efficient algorithm for state-to-state quadrocopter trajectory generation and feasibility verification. *IEEE International Conference on Intelligent Robots and Systems*, pages 3480–3486, 2013. ISSN 21530858. doi: 10.1109/IROS.2013.6696852.

[19] S G Pierce, K C Burnham, D Zhang, L Mcdonald, C N Macleod, G Dobie, R Summan, and Mcmahon D. Quantitative inspection of wind turbine blades using UAV deployed photogrammetry. Technical report, uropean Workshop on Structural Health Monitoring, Glasgow, 2018. URL http://www.ndt.net/?id=23360.

[20] Gauthier Rousseau, Cristina Stoica Maniu, Sihem Tebbani, Mathieu Babel, and Nicolas Martin. Quadcopter-performed cinematographic flight plans using minimum jerk trajectories and predictive camera control. In *2018 European Control Conference, ECC 2018*, pages 2897–2903. IEEE, 6 2018. ISBN 9783952426982. doi: 10.23919/ECC.2018.8550309.

[21] Björn E. Schäfer, Davide Picchi, Thomas Engelhardt, and Dirk Abel. Multicopter unmanned aerial vehicle for automated inspection of wind turbines. *24th Mediterranean Conference on Control and Automation, MED 2016*, pages 244–249, 2016. doi: 10.1109/MED.2016.7536055.

[22] João Paulo Silva and Christophe De Wagter. Quadrotor Thrust Vectoring Control with Time and Jerk Optimal Trajectory Planning in Constant Wind Fields. *Unmanned Systems*, pages 1–23, 2017. ISSN 2301-3850. doi: 10.1142/s2301385018500024.

[23] Ewoud J. J. Smeur, Qiping Chu, and Guido C. H. E. de Croon. Adaptive Incremental Nonlinear Dynamic Inversion for Attitude Control of Micro Air Vehicles. *Journal of Guidance, Control, and Dynamics*, 39 (3):450–461, 3 2016. ISSN 0731-5090. doi: 10.2514/1.G001490. URL http://arc.aiaa.org/doi/10.2514/1.G001490.

[24] Vahram Stepanyan and Kalmanje Krishnakumar. Estimation, Navigation and Control of Multi-Rotor Drones in an Urban Wind Field. *AIAA Information Systems-AIAA Infotech @ Aerospace*, 2017. doi: 10.2514/6.2017-0670. URL http://arc.aiaa.org.

[25] Martin Stokkeland, Kristian Klausen, and Tor A. Johansen. Autonomous visual navigation of Unmanned Aerial Vehicle for wind turbine inspection. *2015 International Conference on Unmanned Aircraft Systems, ICUAS 2015*, pages 998–1007, 2015. doi: 10.1109/ICUAS.2015.7152389.

[26] O Yon Stryk and R Bulirsch. Direct and indirect methods for trajectory optimization. Technical report, Mathematisches Institut, Technische Universitat Munchen, 1992. URL https://link.springer.com/content/pdf/10.1007%2FBF02071065.pdf.

[27] Laszlo Techy and Craig A. Woolsey. Minimum-Time Path Planning for Unmanned Aerial Vehicles in Steady Uniform Winds. *Journal of Guidance, Control, and Dynamics*, 32(6):1736–1746, 11 2009. doi: 10.2514/1.44580. URL http://arc.aiaa.org/doi/10.2514/1.44580.

[28] Nguyen Khoi Tran, Eitan Bulka, and Meyer Nahon. Quadrotor control in a wind field. *2015 International Conference on Unmanned Aircraft Systems, ICUAS 2015*, pages 320–328, 2015. doi: 10.1109/ICUAS.2015.7152306.

[29] Pedro Velez, Novel Certad, and Elvis Ruiz. Trajectory Generation and Tracking Using the AR.Drone 2.0 Quadcopter UAV. In *Proceedings - 12th LARS Latin American Robotics Symposium and 3rd SBR Brazilian Robotics Symposium, LARS-SBR 2015 - Part of the Robotics Conferences 2015*, pages 73–78. IEEE, 10 2016. ISBN 9781467371292. doi: 10.1109/LARS-SBR.2015.33. URL http://ieeexplore.ieee.org/document/7402144/.

[30] L J Vermeer, J N Sˆrensen, and A Crespo. Wind turbine wake aerodynamics. *Progress in Aerospace Sciences*, 39:467–510, 2003. doi: 10.1016/S0376-0421(03)00078-2. URL https://www.sciencedirect.com/science/article/pii/S0376042103000782.

[31] Xingyu Xiang, Zhonghai Wang, Zijian Mo, Genshe Chen, Khanh Pham, and Erik Blasch. Wind field estimation through autonomous quadcopter avionics. *AIAA/IEEE Digital Avionics Systems Conference - Proceedings*, 2016-Decem:1–6, 2016. ISSN 21557209. doi: 10.1109/DASC.2016.7778071.

[32] Hao Xu, Xiangrong Xu, Yan Li, Xiaosheng Zhu, Liming Jia, and Dongqing Shi. Trajectory planning of Unmanned Aerial Vehicle based on A* algorithm. In *4th Annual IEEE International Conference on Cyber Technology in Automation, Control and Intelligent Systems, IEEE-CYBER 2014*, pages 463–468. IEEE, 6 2014. ISBN 9781479936687. doi: 10.1109/CYBER.2014.6917508. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6917508.

3

Article

The article as finished in January 2020 is presented in this chapter. The article presents a robust trajectory generator for wind turbine inspection which takes into account wind and is able to online create safe emergency routes. The used approach and experimental setup are explained, followed by the results and conclusions. For the thesis a provided model and controller are used as explained in the article. However this created some issues which can be related to (a combination of) the used controller and drone model. When simulating a constant wind field the drone tends to perform satisfactorily, however when flying with 2 (+/- 0.3) meters per second headwind the drone crashes. Increasing the wind speed to up to 8 meters per second or below 2 meter per second does not cause this. All tests for the article were performed around 7 meters per second wind, hence this issue does not return in the article.

Trajectory generator for automated wind turbine inspector drones

L. Gooijaers*
Delft University of Technology, P.o. Box 5058, 2600GB Delft, Nederland

To be able to increase the safety and efficiency of automated wind turbine inspection a robust solution is needed that takes into account wind during trajectory planning. This paper presents such a solution in the form of an algorithm, which uses Dijkstra's algorithm as a basis for optimisation. To allow for online replanning the horizontal and vertical trajectory planning are separated, creating a computationally cheap problem. The solution is based on an in advance known static environment, hence no time and resources are spend before the inspection. Three experiments show that the presented algorithm is safe and is able to take into account wind during emergency procedures. In addition it is shown that it is able to safely replan when the wind changes during an emergency. These results show that the proposed algorithm can serve as a viable basis for a safe and efficient solution for automated wind turbine inspection.

I. Nomenclature

c	=	path cost
K_{turb}	=	wind turbine wake constant
$L_{(vis)Wp}$	=	List of (visible) waypoints
$M_{adj}, M_{wind}, M_{dist}$	=	Adjacency matrix, wind matrix, distance matrix
pos_{xy}	=	xy position
$sBox$	=	Wind turbine safety box
T, T_{xy}, T_z	=	Trajectory, horizontal trajectory, height profile
$\bar{V}_{wind,xy}$	=	Adjacency matrix based on average wind speed
V_{max}	=	Maximum drone speed
$wp_{pos}, wp_{home}, wp_{safe}$	=	current waypoint, home waypoint, safe waypoint

II. Introduction

The use of Unmanned Aerial Vehicles (UAV) for inspection of difficult to access structures has increased over the last years[1]. One of the applications is using multi-rotor vehicles for inspection of wind turbines. Already several companies offer solutions to inspect wind turbines with drones, either remotely piloted such as AetosDrones, IAS or ABL Drones or autonomously such as SkySpecs. Procedures involving remote pilots are vulnerable to human error and limited by the capabilities of the UAV pilot. Autonomous inspection does not have this drawback, however the challenge remains to guarantee safe and robust operations while staying efficient [2].

The research on wind turbine inspection mostly focuses on the use of vision as a basis of automated navigation which is an extensively studied area for UAV's in general. Stokkeland, Klausen and Johansen [3] present a solution for wind turbine inspection using the Hough transform for detection of the wind turbine and a Kalman filter for tracking. Høglund [4] presents a similar solution and combines this with the use of optical flow. For both solutions it is required that the drone first finds the wind turbine hub to calibrate its position. As an alternative with increased accuracy and robustness a 2D LiDaR based mapping solution is presented by Schäfer et. al. [5]. Before inspection a multi-rotor vehicle scans the wind turbine park which is used to create a map. The inspection vehicles use on-board LiDaR sensors to navigate throughout the park. The downside of both the vision and LiDaR based solutions is the increase in operating time due to calibrating or mapping procedures.

*MSc Student, Faculty of Aerospace Engineering, Control and Simulation Department, Delft University of Technology.

Besides the basis of navigation there are several possible methods for trajectory generation. For trajectory generation it is possible to take the vehicle dynamics in consideration[6]. The most common solutions that do this are based on a boundary value problem, numerical optimisation or path re-parametrization. Methods based on boundary value problem are reliable and can run real-time by sequentially solving a series of boundary value problems [7][8]. However applications are only on a small scale and aerodynamic effects are ignored. For numerical optimization methods direct or indirect expressions are used to optimise for parameters such as time or total energy consumption[9]. These methods can generate almost optimal solutions for UAV trajectory generation problems, using piecewise polynomials or splines as a basis function[10],[11]. They are suitable for a dense environment, taking into account corridor constraints or changing environments,[12],[13]. The downside is that such an implementation is computationally expensive for larger trajectories. Lastly path re-parametrization methods first create a pure geometric path using primitives such as lines, polynomials or splines and ensure dynamic feasibility by building a motion profile around this geometric path[14]. Due to the relative simple geometric paths it better suitable for application with longer trajectories, however it can be less optimal in comparison to numerical optimization methods.

When vehicle dynamics do not have to be taken into account simpler methods of generating UAV trajectories can be used by applying path planning algorithms, such as graph search (Dijkstra or A*[15]) or random search (RRT*) [16]. The resulting trajectories are often not feasible due to sharp corners, hence the controller should be able to handle this. Due to the simplicity of these methods fast replanning is feasible, hence it is suitable for online replanning of long trajectories. However the trajectories created by these methods are less optimal in comparison to the previously mentioned methods.

An important factor to take into account during trajectory optimisation for wind turbine inspection is wind. Both onshore and offshore wind effects are important whereby the latter has higher but more consistent wind speeds[17]. When a wind turbine park is still partially operational during inspection it can be beneficial to also take wind turbines wakes in consideration[18]. To be able to take into account known winds both Guerrero, Escareno and Bestaoui [19] and Stepanyan and Krishnakumar[20] present optimization methods combined with a control strategy to reduce excessive energy consumption. Both methods have the drawback of being computationally expensive.

This work is part of a project which has as goal to provide a safe and efficient solution for automated wind turbine inspection. This project includes all steps from creating a database with the lay-out of available wind turbines to the processing of measurements taken with automated UAV's. The focus of this paper will be on trajectory generation, which is a key component to ensure that the solution is safe and efficient.

The contribution of this research is a robust trajectory generator for wind turbine inspections which takes into account wind and is able to online create safe emergency routes. The procedure is based on an in advance known static environment, hence no time and resources are spend before the inspection. The method is robust and safe by using Dijkstra's algorithm, optimising for travel time and taking into account wind, while being computationally cheaper than other solutions.

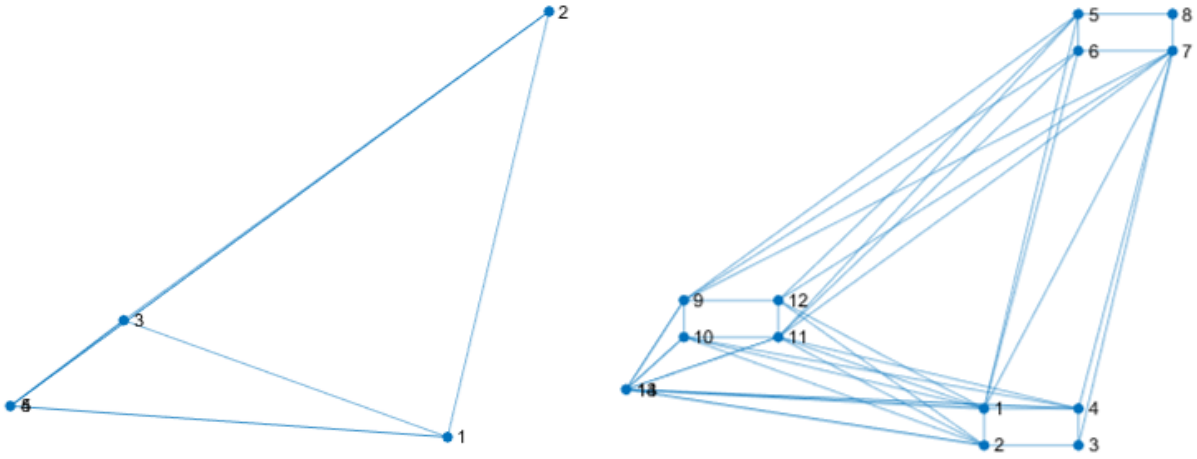
Previous research and used optimization procedures are explained in section III. After this the experimental setup is presented in section IV. The results of the experiments and their discussion are provided in section V. Lastly the conclusion and recommendations are given in section VI.

III. Methodology

In this section the direction of this exploratory research on wind turbine inspection is explained. First the nominal operation procedures are explained in short in section III.A. This is followed by the emergency operation procedures in section III.B. Finally the implemented approach is provided in section III.C

A. Nominal operation

For nominal path planning two optimization steps are implemented. The first step optimises the order in which the turbines are visited while the second step creates a height profile. The path around the wind turbine is fixed and is determined by a large set of waypoints which indicate the coordinates at which a picture of the turbine has to be taken. The turbine order is determined by solving a travelling salesman problem using a visibility graph which takes into account wind. An example of such a graph is given in figure 1a with three turbines (node 1,2 and 3) and a start and finish waypoint that coincide (node 4,5). The lay-out of the visibility graph is determined before the run and only the weights of each link have to be updated during the inspection when the wind changes.



(a) Nominal operation **(b) Emergency replanning**

Fig. 1 Examples of the visibility graphs as used during trajectory planning

B. Emergency operation

The main focus of this paper is to be able to safely handle an emergency during wind turbine inspection, such as hardware failure, rapid battery discharge or unexpectedly high windspeeds. When a failure occurs a new trajectory must be determined, taking into account obstacles, wind and possibly battery life. The drone should be able to take the safest route back to the home station. The safest route in this case is defined as the fastest route back to the home station in which the drone does not enter a safety zones.

To ensure a low computational load, allowing fast replanning, the trajectory optimization in the horizontal plane is done first, after which a height profile is created. This creates a less optimal solution, however it greatly decreases the complexity and increases the robustness of the solution. The optimization in the horizontal plane is done using Dijkstra's algorithm, while for this research the height profile consist of a constant gradual descent.

1. Dijkstra's algorithm

The safest way to the home station is determined using the Dijkstra algorithm. Although the first Dijkstra's algorithm is published in 1959[21] the application of this algorithm is still very useful and relevant[22][23]. The A* algorithm is computationally faster since it uses heuristics[15], which unlike Dijkstra's algorithm is not guaranteed to give the global optimum. Due to the relatively low amount of waypoints using A* will have little benefit over Dijkstra's algorithm in terms of computational cost. Using Dijkstra's algorithm ensures the safest possible trajectory based on the adjacency matrix since no heuristics are used.

Dijkstra's algorithm makes use of a (di)graph (topological form) of which the data can be stored in an adjacency matrix (matrix form). An entry in the adjacency matrix Adj_{xy} corresponds to the cost of travelling from node x to node y. When the direction in which the link is traversed has no influence on the cost, the matrix is symmetric. For this research the entries of the adjacency matrix are determined by the time it takes to traverse the link. Since the drone dynamics are not taken into account this is determined by dividing the link distance by the maximum drone speed.

An example of a 2D graph as used for Dijkstra's algorithm is shown in figure1b. The nodes of the graph are created by choosing the nodes as the corners of the wind turbine safety boxes, the home waypoint and the current position of the drone. In this example three wind turbines are used and the start and end coordinates coincide. The safety boxes ensure that the drones never fly too close too the wind turbines.

To show the working of the Dijkstra algorithm a directional graph of figure 1a is shown in figure 2 and the corresponding adjacency matrix in shown in equation 1. A directional graph or digraph implies that the cost for traversing a link depends on the travel direction over that link. This way directional effects that impact the cost can be taken into account, such as the wind field. For this research the link cost are defined as the travel time over that link, hence links in the same direction as the wind are cheaper. In this example the wind speed is assumed to be the same as travel speed. This makes link pairs $(3 \rightarrow 4)$ and $(2 \rightarrow 3)$ infeasible since they are directly opposite to the wind direction.

Dijkstra's algorithm keeps a list of all unvisited nodes and initialises a cost list with the cost to reach all nodes and sets these to infinity. Only the cost of the starting node is set to zero and this node is taken as current node. After this an iterative process starts where the current node changes, but the calculation stays the same. For the current node all unvisited reachable nodes are determined and the cost to go there is updated in the cost list. Next the current node is set to visited. The next current node is determined by taking the node with the lowest cost to travel to and contained in the list with unvisited nodes. When the current node is the same as the goal node the process stops, else the procedure is repeated. This algorithm is explained in more detail in the pseudocode in appendix A.

As an example in the case of figure 2 the *unvisited* list is initialised containing the nodes 1 to 4. In the case where the goal is to travel from 2 (*start node*) to 4 (*destination*), the *current node* is first set at 2. Each node in row 2 of the adjacency matrix in equation 1 is now checked for the cost. Only one link is available (only one non-zero entry) and that one is added to a cost list (total cost is 11) and node 1 is set as the new *current node*. Repeating this would give a route cost of 17 to node 3 and 21 to node 4. Next node 3 would be the *current node*, resulting in nothing new since the cost to travel to node 2 is already set at 0. Hence node 4 is set as *current node* after which the algorithm stops and returns a total cost of 21 for the shortest path.

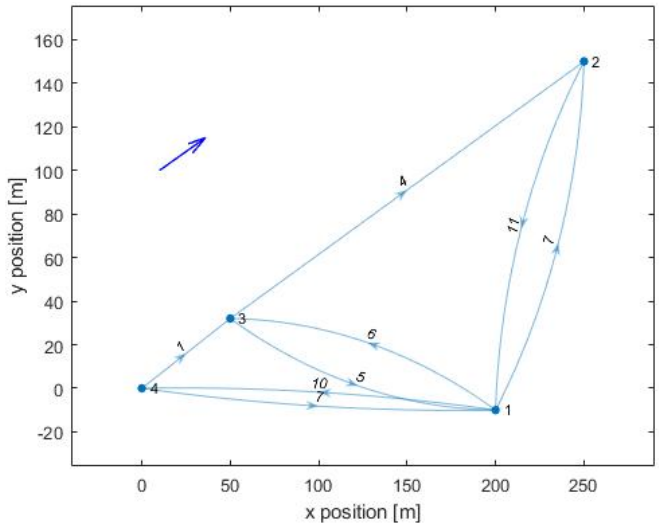


Fig. 2 Example of a directional graph based on figure 1a. The displayed edge costs are dependent on the wind direction as indicated by the blue arrow. The corresponding adjacency matrix is shown in equation 1.

$$Adj_{xy} = \begin{bmatrix} 0 & 7 & 6 & 10 \\ 11 & 0 & 0 & 0 \\ 5 & 4 & 0 & 0 \\ 7 & 0 & 1 & 0 \end{bmatrix} \quad (1)$$

2. Wind field

The use of an adjacency matrix allows for the wind field to be included in the optimization process without greatly impacting computational efficiency. Due to the lack of high structures around wind turbine parks the wind field is relatively constant. The two most important influences on the wind field are the wind turbine wakes and the wind gradient. Since the optimization in the horizontal plane and vertical plane are done separately the wind field determination is also separated. In the horizontal plane the wind gradient has no influence since it only change with height. The wind turbine wake will influence both horizontal and vertical wind optimization. The wind turbine wake is complex and high-fidelity models have been created[18], however a simple model such as created by Bastankhah and Porté-Agel[24] can suffice. For this research it is assumed that the wind turbines are non-operational during inspection, hence no wind turbine wakes have to be taken into account.

C. Approach

To be able to achieve save emergency replanning the algorithm as shown in algorithm 1 is implemented. Whenever an emergency occurs the algorithm is used to determine a new trajectory. If the drone is currently inspecting a wind

turbine, hence is inside a safety box, the first step is to fly outside the safety box towards the safe waypoint (wp_{safe}), which is determined by **ClosestWp**. The first waypoint outside the safety box is used to determine all visible waypoints which are then added to the distance adjacency matrix. The distance adjacency matrix (M_{dist}) contains the distances of all valid links between waypoints. When the drone is in between wind turbines the current position is immediately added to the distance adjacency matrix.

Secondly the wind adjacency matrix (M_{wind}) is determined and is used to save the estimated ground speed for each link. Each link cost is calculated by subtracting the average wind speed per link from the maximum drone speed. To be able to take into account wind turbine wakes the cost can be multiplied by a constant K_{turb} in range $[0, 1]$ which can compensates for the turbulence. After this the adjacency matrix (M_{adj}), used for the optimization, is determined by dividing the distance of each link by the estimated ground speed, giving a estimated travel time. Lastly Dijkstra's algorithm is used to determine the horizontal trajectory based on the adjacency matrix, starting waypoint and final waypoint.

Algorithm 1: Emergency replanning

Result: Updated trajectory T
if *emergency triggered* **then**
  if *inside safety box* **then**
    wp_{safe} = **ClosestWp**($pos_{xy}, sBox$);
    L_{visWp} = **Raycasting**($wp_{safe}, L_{wp}, sBox$);
    Add (wp_{safe},L_{visWp}) to M_{dist};
  else
    L_{visWp} = **Raycasting**($pos_{xy}, L_{wp}, sBox$);
    Add (pos_{xy},L_{visWp}) to M_{dist};
  end
  foreach *entry of* M_{wind} **do**
    $M_{wind,xy} = (V_{max} - \bar{V}_{wind,xy}) * K_{turb}$;
  end
  $M_{adj} = M_{dist}/M_{wind}$;
  $[T_{xy}, c]$ = **Dijkstra**($M_{adj}, wp_{pos}, wp_{home}$);
  Optimise T_z;
  Set new T;
end

To be able to add a node to the graph the visible waypoints, hence feasible links, have to be determined. This is done by running the raycasting algorithm, which determines a list of nodes that are visible from a certain point. This is done by creating a line segment from the start node to each other node. Each segment is checked against each turbine safety box segment for intersections. When an intersection occurs it implies that the node is not visible. The pseuso-code of the raycasting algorithm is given in appendix B in algorithm 3.

IV. Experimental setup

In this section the simulation model is explained in section IV.A. In section IV.B the parameters used during three experiments are presented. The three experiments are used to test the safety and robustness of the proposed implementation. Experiment one and two test the emergency planning for respectively a three and nine turbine scenario, while experiment three is performed to show the replanning capabilities after a wind change. All simulations are performed using Simulink on a desktop running windows 10, using a Intel Core i5-6600 CPU and 16,0 GB RAM.

A. Simulation model

To quadrotor used in all experiments is the Parrot Bebop 2, which has a maximum speed of 16 meters per second, a weight of 500 grams and a maximum flight time of 25 minutes*. The model used to simulate this drone is developed by Sihao[25] and is a high-fidelity simulator, which simulates aerodynamic effects.

*https://support.parrot.com/nl/en/support/products/parrot-bebop-2

The controller consist of a three-loop nonlinear controller and is proven to be useful for high-speed flight using the Parrot Bebop2 quadrotor. Although wind turbines inspection happens in a different environment, using different controller inputs, the model is able to cope with both aggressive manoeuvres and high speed flight. For the tests the quadrotor will no longer fly aggressive manoeuvres in a small environment, however high speed flight is still desirable over long distances.

B. Experiment parameters

A Monte-Carlo simulation will be used for all experiments to test the working of the algorithm. These experiments will only focus on the time after a failure has happened, not on nominal flight. The lay-out of the wind turbines is based on the Cornwerd windpark (53°05'34.8"N 5°23'31.2"E). The size of the wind turbines used in the experiments is smaller due to the Bebop 2 being much smaller and lighter than drones that will be used during an inspection. The used mast height, hub depth and blade length are respectively 40, 5 and 20 meters in size. Each turbine blade is inspected on two side and an example of such a trajectory is shown in figure 3. In this figure two types of boxes around the wind turbine are shown. The smaller boxes are used as guidance during the inspection, while the larger safety boxes around the entire wind turbine are used for navigation through the park. For all experiments all wind turbines are standing still, hence no wind turbine wake is taken into account.

Experiment one will have two variables, namely the time of failure and the wind direction. The wind direction is expected to have little to no impact on the trajectory generation when the quadrotor flies in the same direction and small wind changes will also have very little impact. Therefore only a limited number of different directions is tested. For each wind direction 800 runs are performed, evenly distributed over the failure time range. The number of runs that can be performed is limited due to the limitation in computational resources. The lowest failure time is chosen as the time it takes to complete the first straight link. The highest failure time is the average maximum flight time to inspect all wind turbines.

Table 1 Experiment variables

Nr	Variable	Test values
1	# Turbines [-]	3
	Wind direction [°]	-180, -120, -90, -60, 0, 90
	Failure time range [s]	<200, 4300>
2	# Turbines [-]	9
	Wind direction [°]	-120, -90, -60, 0
	Failure time range [s]	<200, 13000>
3	# Turbines [-]	3
	Wind direction [°]	-120, -90, -60, 0
	Wait time range [s]	<50, 500>

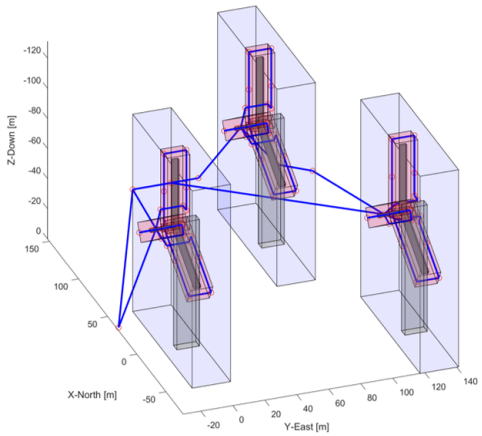


Fig. 3 A nominal trajectory, including wind turbine safety boxes.

To make the behaviour of the algorithm more apparent experiment one is repeated, but with a smaller set of test values and an increased number of wind turbines. A smaller set of test values is needed due to the limited computational resources, while more wind turbines increase the computational complexity. The used wind direction values are those at which the most effect is expected.

Lastly the third experiment tests the effect of a wind change during an emergency. Similar to the wind directions in experiment one and two, only a limited number of wind changes will be tested. The time it takes before the wind changes after an emergency (wait time) is varied as shown in table 1. The lowest value is chosen such that the quadrotor has some time to start following its first emergency trajectory, while the highest value is chosen as the average time after which the route will not change anymore. For all cases a single failure time of 2000 seconds is used, at which the first emergency is initiated.

For experiment one the nominal maximum drone speed is assumed to be 16 [m/s] while the maximum drone speed after failure is set at 7 [m/s]. The wind speed is chosen to be 7 [m/s] as well to ensure the drone is unable to fly straight into the wind and has to find other paths. For the second experiment the maximum drone speed after failure is set at 7.7

[m/s], 10 % higher than the wind speed of 7 [m/s]. Therefore flying straight into the wind will be very disadvantageous however it is still possible. This makes it possible to better compare Dijkstra's algorithm with and without wind, since all links are still feasible. Lastly for all experiments a wind noise is used of +/- 0.5 [m/s].

V. Results and Discussion

In this section the results of each experiment are presented and discussed. All figures in this section only show the trajectories after a failure occurs, hence no nominal flight is included. Some examples of such trajectories are shown in figure 5. In this figure the 3 wind turbine safety boxes are shown together with the home waypoint to the left at coordinate (0,0,0) [m]. The four trajectories are created in a zero wind situation and at different failure times during inspection. To be able to discus the results the numbering of the wind turbines as shown in figure 4 is used. In this figure a top view of the used wind turbine park is shown, together with the nine numbered safety boxes and the home waypoint.

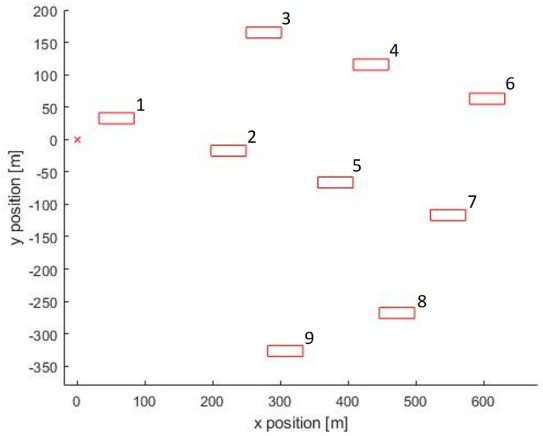


Fig. 4 Wind turbine numbering convention

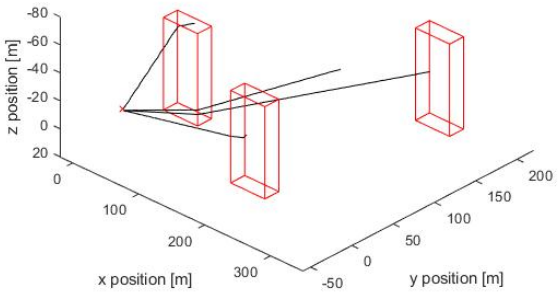


Fig. 5 Example trajectories after failure for 3 turbines

A. Experiment 1 and 2: Single wind directions and varying failure times

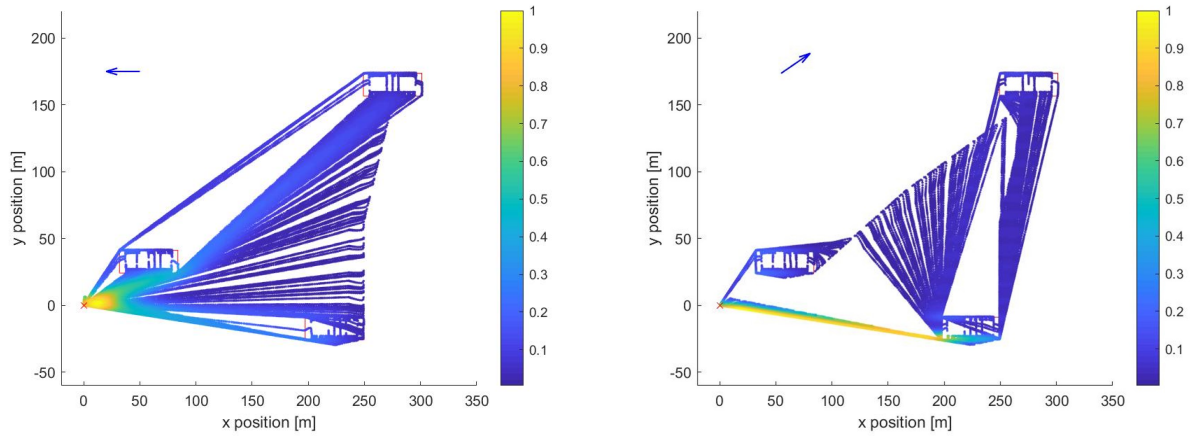
For the first and second experiment a failure is triggered at different times during inspection for respectively six and four different wind directions. For each run the drone is able to successfully return to the home waypoint, however via different routes based on the wind condition. The results for each run are shown in heatmaps in figures 6a to 6e. Each heatmap is determined using the spatial density of the position data of the runs, where the colour in the heatmap is determined by the normalised probability density estimate. Therefore the heat scale gives an indication of the chance that the drone passes through a point after failure. The position data is gathered by taking a sample of the position at each second during all the runs. For all cases it can be seen that several flights start inside the wind turbine safety box, since they are triggered during the inspection. This can be seen as the short vertical lines inside the safety boxes.

Figure 6a shows a run in which the wind direction is 270°, hence to the west, as indicated by the arrow in the figure. In this case the data shows that the drone takes the shortest route, equivalent to a zero wind scenario. This is as expected since the wind acts always in roughly the same direction as the drone, it will always have a positive effect on the shortest routes to the home waypoint.

In the cases where the wind acts in the opposite direction to the general flight direction of the drone a more clear effect of the used algorithm is shown. In the case where the wind direction is 60°(figure 6b) the wind acts almost opposite to the link between the turbine one and two, hence this link is not used unless the failure happens very close to turbine one. In the other cases it is more beneficial to fly via turbine three, which is shown by a high density of flights between the third turbine and the home waypoint.

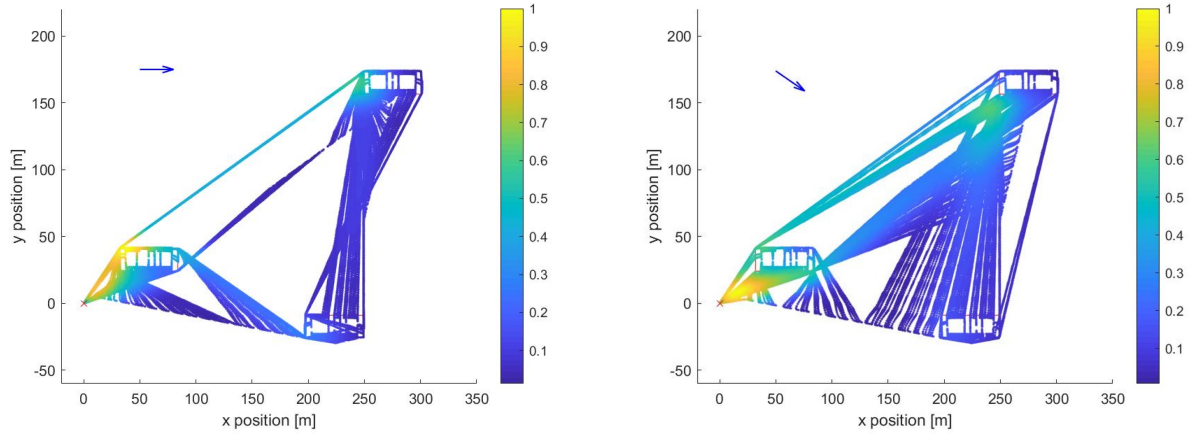
A similar effect is shown in the cases where the wind direction is 120°(figure 6d) and 90°(figure 6c). In figure 6d it is clear that the link between the third turbine and the home waypoint/wind turbine one is almost unused, while most emergency routes are planned via turbine two. In figure 6c all flight directions are in the opposite direction of the wind hence routes are chosen that are more perpendicular to the wind.

For the cases where the wind direction is perpendicular on the general flight direction (0°and 180°) only small effects can be seen, as shown in figures 6e and 6f respectively.



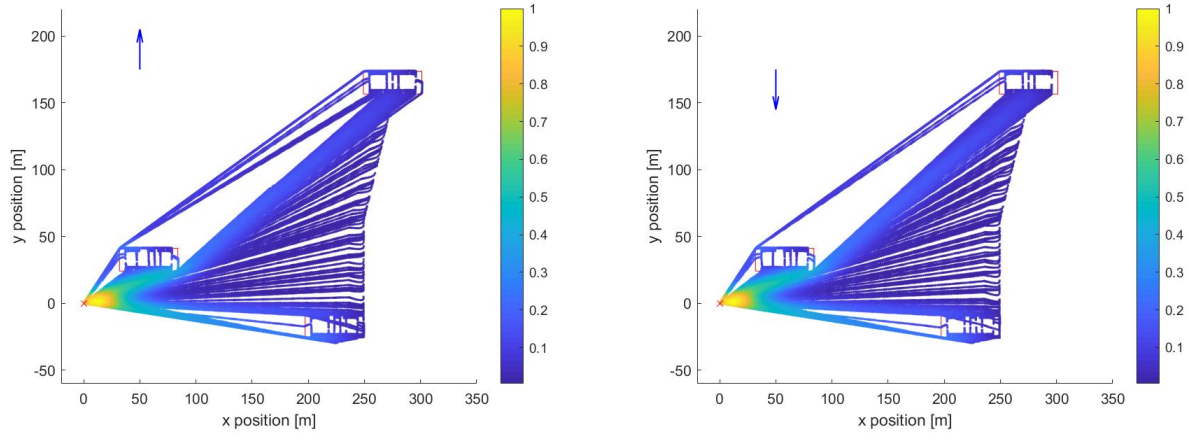
(a) Wind direction: 270°

(b) Wind direction: 60°



(c) Wind direction: 90°

(d) Wind direction: 120°



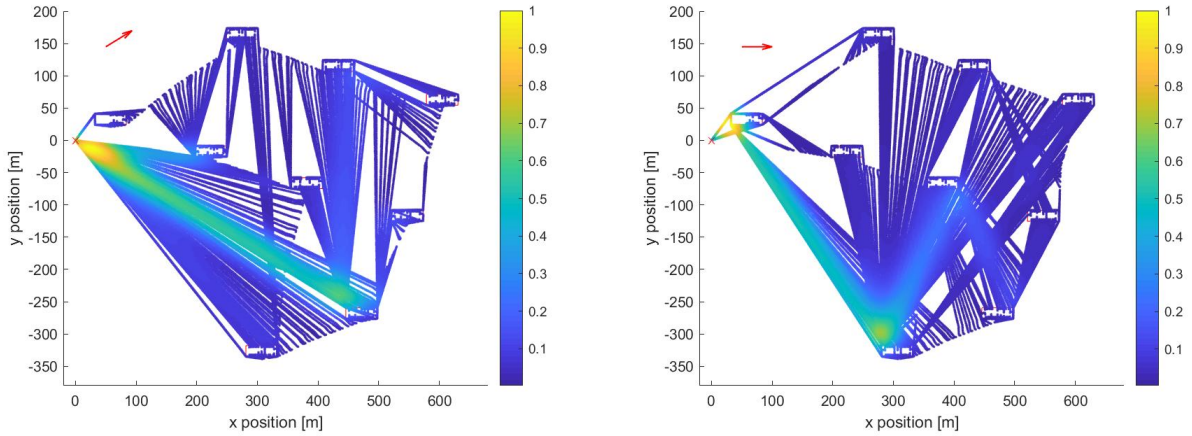
(e) Wind direction: 0°

(f) Wind direction: 180°

Fig. 6 Heatmaps of the scenarios for 3 turbines as part of experiment 1. Each heatmap shows only the trajectory data after a failure (emergency rerouting) and combines 800 runs for a single wind direction (indicated by the blue arrow). The colour of the heatmap shows the probability of encountering the drone at a certain point on the map.

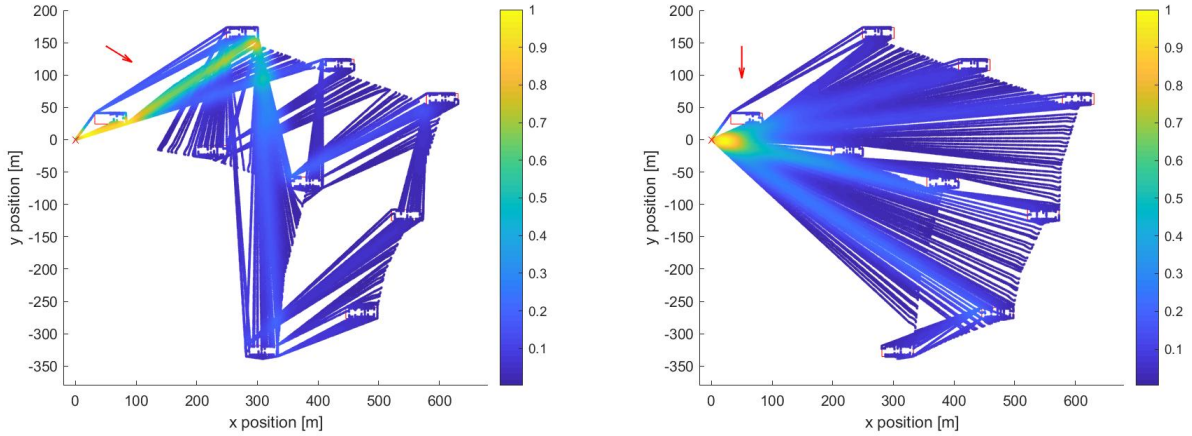
To test the algorithm with a more realistic scenario experiment two repeats experiment one, however using nine wind turbines instead of three. Instead of all six previously used wind directions only the three directions are used which previously yielded the most clear results. In addition one wind direction perpendicular to the general flight direction is used. All results are shown in figures 7a to 7d.

The results are similar to experiment one, however they show a more clear general behaviour. In all cases the preferred flight direction is almost perpendicular to the wind direction, resulting in a zigzag pattern toward the home waypoint. This is a results of the wind speed being 90% of the maximum drone speed during emergency.



(a) Wind direction: 60°

(b) Wind direction: 90°



(c) Wind direction: 120°

(d) Wind direction: 180°

Fig. 7 Heatmaps of the scenarios for 9 turbines as part of experiment 2. Each heatmap shows only the trajectory data after a failure (emergency rerouting) and combines 800 runs for a single wind direction (indicated by the red arrow). The colour of the heatmap shows the probability of encountering the drone at a certain point on the map.

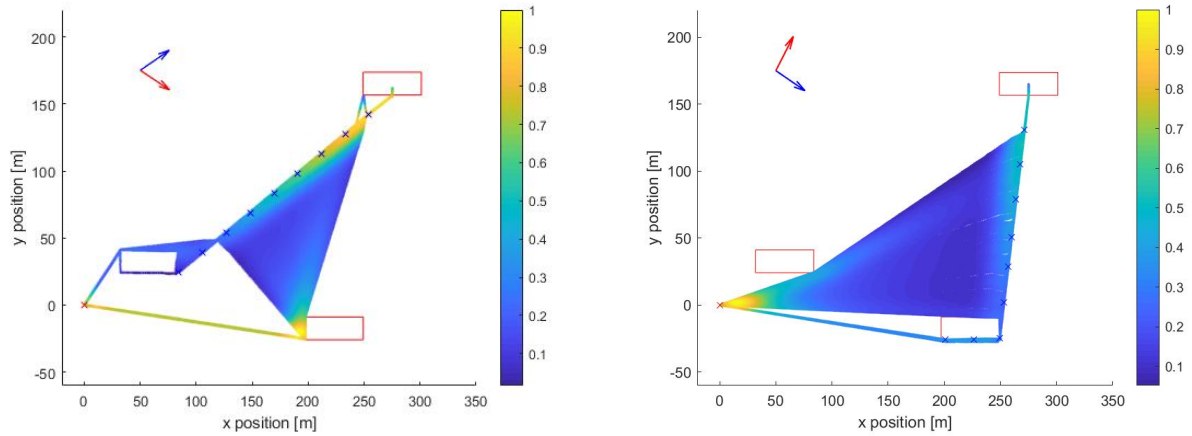
B. Experiment 3: Varying wind direction and wait times

The third experiment shows the result of a wind change during an emergency. For a single failure time of 2000 seconds four scenarios are run using the same starting wind conditions as experiment two. For the 60°and 120°a wind change of 60°is simulated resulting in the respectively a wind direction of 120°and 60°as can be seen in figures 8a and 8b. Similarly the wind changes are simulated between 0°and 90°as shown in figures 8c and 8d. In each sub-figure of figure 8 a single emergency route is followed until a wind change happens. The blue crosses coincide with this original

emergency route and mark the replanning location of every 100^{th} run. The blue and red arrow indicates the initial and changed wind direction respectively.

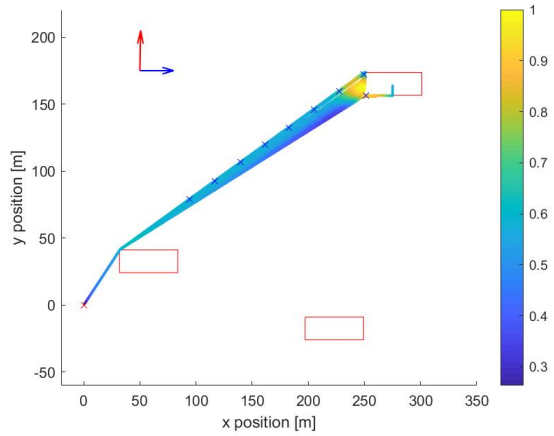
The most clear result are again seen at the routes which are in the opposite direction of the wind direction (60°and 120°). In figure 8a The algorithm replans the emergency route via the third turbine to avoid flying almost straight into the wind. Only when the drone is almost at wind turbine one it does not fly back to wind turbine three, however the drone does change its route to fly around wind turbine one. Similarly in figure 8b the original route flies around wind turbine three, however replanning allows for a path straight to the home waypoint.

The changes for the runs with cardinal wind directions is smaller but also clear. In figure 8c the original path goes through the upper left corner of wind turbine two to decrease the head wind. After replanning the drone avoids this point up until the moment when the drone has already passed this point, after which no difference is noticeable. The opposite happens in figure 8d where the original path is almost a straight line to the home waypoint, however after the wind change the first set of runs replans the path to go via the upper left corner of wind turbine two. This is done to avoid the link between the lower right corner of wind turbine one and the home waypoint, which is almost aligned with the wind.

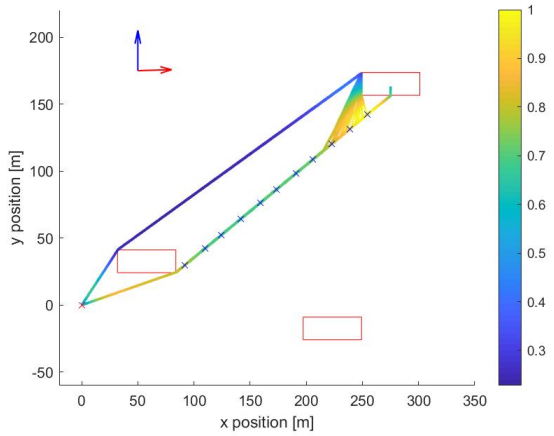


(a) Wind direction 60°to 120°

(b) Wind direction 120°to 60°



(c) Wind direction 90°to 0°



(d) Wind direction 0°to 90°

Fig. 8 Heatmaps of the scenarios for 3 turbines as part of experiment 3. Each heatmap shows only the trajectory data after a failure at 2000 seconds (emergency rerouting) and combines 800 runs for a single wind direction (indicated by the blue arrow) and one wind change (indicated by the red arrow). The crosses indicate each 100^{th} rerouting position due to the wind change at different times. The colour of the heatmap shows the probability of encountering the drone at a certain point on the map.

C. Performance

To assess the performance of the presented algorithm the used cost metric (travel time) is compared to the theoretical cost as determined by Dijkstra's algorithm without taking into account wind. For experiment one and two the comparison is shown in the boxplots in respectively figures 9 and 10. The data in this figures is corrected since the estimates as made by the algorithm were significantly to low, as can be seen in table 2. The most important reason for this is the restrictions due to the controller and how these are handled. When a link is to long the input to the drone is to large causing it to spin out of control. Setting limits on the velocity and acceleration references did not solve this, therefore each long distance link is broken up into several smaller links for control purposes. This causes the drone to accelerate and decelerate almost constantly during flight between turbines, giving a very low average speed in comparison to the used maximum drone speed. The travel time estimates also do not include any vehicle dynamics which, together with the controller issues, explain the variances and outliers.

The correction factor is determined by the difference between the mean value of the estimated travel time and actual travel time. The factor for the cost differences in figure 9 is highly depend on the wind direction, namely varying between 3.7 for the 90°case and 27.8 for the 270°case. The high factor for tailwind is due to the fact that the controller decelerates the drone at each waypoint to the same ground speed independent of wind speed. Hence in the case of tailwind the deceleration is much higher than in the case of headwind, where the ground speed is lower to begin with. The large differences in cost estimation when the wind direction and flight direction are aligned is also reflected by the large amount of outliers, on which the only exemptions are the 0°and 180°case. These exemptions can be attributed to the wind being perpendicular to the flying direction, hence having the least effect on the results.

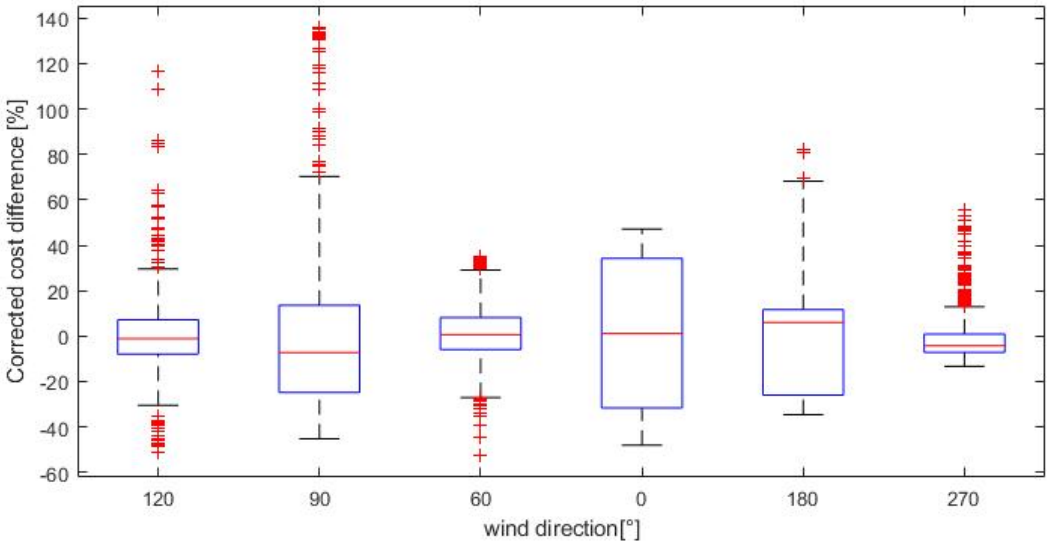


Fig. 9 Boxplots for experiment 1, showing the difference between the calculated and actual travel time, corrected for the difference in mean between both. Each boxplot shows in single wind direction and gives an indication of the variance per wind direction.

The more realistic case with nine turbines present a more constant image with much less outliers (figure 10) and more constant correction factors for difference between the actual and computed travel times (table 2). In the 90°case the boxplot shows only outliers in the lower part, opposite of the case with three turbines. The correction factors for 60°, 90°and 120°are the same, which can be explained by the very similar (zigzagging) behaviour in all cases combined with longer links in comparison to experiment one. The latter allows the drone to make less turns hence the velocity profile is more constant. Also the graph grid is much larger allowing for better optimization. Experiment one with three turbines allows for clear visualisation of the behaviour of the algorithm, however it might favour some wind directions over other in terms of optimization.

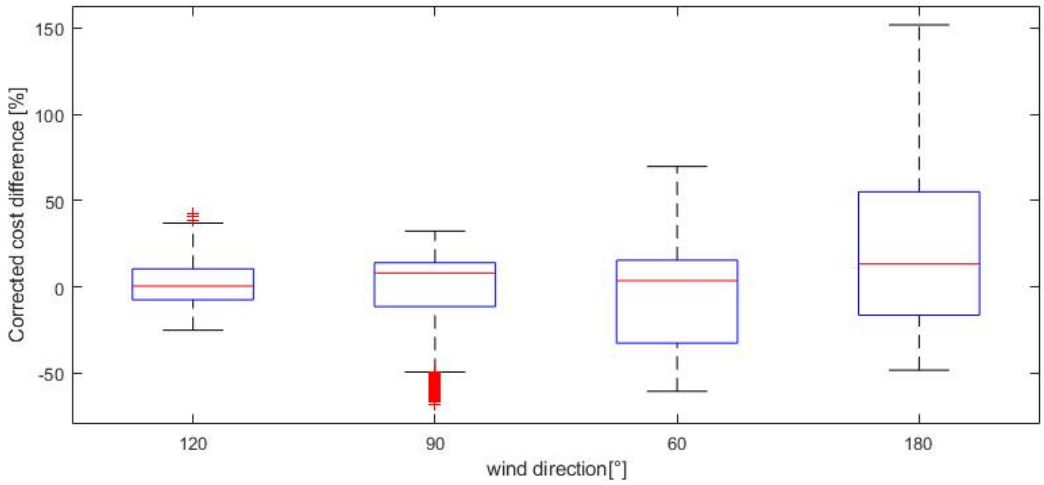


Fig. 10 Boxplots for experiment 2, showing the difference between the calculated and actual travel time, corrected for the difference in mean between both. Each boxplot shows in single wind direction and gives an indication of the variance per wind direction.

Table 2 Differences between the means of the actual and estimated travel times

Wind direction	Mean difference experiment 1	Mean difference experiment 2
0°	10.4	-
60°	4.6	7.4
90°	3.7	7.4
120°	8.4	7.4
180°	18.5	11.8
270°	27.8	-

Besides the accuracy of the algorithm the cost metric is also compared to the case where Dijkstra's algorithm does not take into account any wind effects. The maximum drone speed in experiment one is equal to the wind speed hence all links in the graph opposite to the wind direction are infeasible. When Dijkstra's algorithm does not take into account wind it uses these links hence infeasible paths are generated. For experiment two the maximum drone speed is slightly higher than the wind speed hence no infeasible links are created, allowing for a comparison between both cases. The results are shown in table 3 and show the clear advantage of optimising for wind. For each wind case the wind based Dijkstra's algorithm shows a decrease in travel time. The 180°case shows the least improvement, which can be expected since the optimised routes only slightly differ from the shortest paths. For the other cases the decrease in travel time vary from 43.0% to 74.2%. The variation is caused by the asymmetry of the wind turbine layout, hence certain lay-outs allow for better optimization. The increase in efficiency is achieved by the algorithm which does not take into account aerodynamic effects, while they are included in the simulation. Therefore the algorithm is able to recompute a trajectory in well under one second. The very low computational impact indicates that the algorithm can be used for online replanning and the increase in efficiency is an indication the wind optimization for high-wind scenarios is very important.

Table 3 Theoretical profit of wind optimization for 9 turbines

Wind direction	Average travel time decrease
60°	64.9%
90°	74.2%
120°	43.0%
180°	3.6%

VI. Conclusion

The goal of this research is to create a robust trajectory generator for wind turbine inspections which takes into account wind. Using Dijkstra's algorithm an emergency replanning algorithm is proposed that allows for this. The proposed algorithm is able to replan a route in under one second, while it yields significant decreases in travel times for high-wind scenarios. Experiment one shows that the proposed algorithm is suitable as trajectory generation method for wind turbine inspection and that it allows for wind optimisation. Experiment two confirms this for a more realistic scenario. This experiment also shows an increase in efficiency in comparison to the no-wind optimisation since there is a decrease in travel time in all cases. The most ideal case shows a theoretical decrease in travel time of 74.2%. Lastly the third experiment displays the algorithms capability to take into account wind changes during emergency flight, confirming the robustness of the proposed solution.

The travel time estimations are consistently to low. In the first experiment the differences between the estimated and actual travel time vary from a factor of 3.7 times to low for headwind to 27.8 to low for tailwind. The differences in experiment two are much more consistent with 7.4 times to low in three out of four cases. These underestimations are mainly caused by the implementation of the controller, which is unable to cope with waypoints that are far away. To overcome this each link is divided in smaller sub-links causing the drone to decelerate and accelerate excessively.

For future work it is advised to look into a more realistic travel time estimation, to overcome the underestimation. This can be done by improving the controller and by taking into account that the drone will not always fly at maximum speed. To create a trajectory which is easier to track it can be beneficial to create a velocity profile for each trajectory. It is also recommended to use a more realistic wind scenarios, based on actual wind data or by taking into account wind shear and wind turbine wakes. This also allows for optimization of the height profile. Lastly the graph size can be increased by adding nodes other than the wind turbine safety boxes. Therefore the algorithm is able to create more optimal routes and the efficiency increase is less dependent on the specific lay-out of the wind turbines.

Appendix A - Dijkstra's algorithm

In this appendix the Dijkstra's algorithm [21] as implemented in this research is shown in algorithm 2. In this algorithm the cost to reach each node from the starting node is determined using an iterative process and stored in $costList$. When the goal node is reached the cheapest path and corresponding cost are determined and stored in respectively $sPath$ and c.

Algorithm 2: Dijkstra's algorithm

```
Result: Shortest path (sPath) and corresponding path cost (c)
foundList = startNode;
nrFoundNodes = 1;
nrNodes = total number of nodes;
for i = 1 to nrNodes do
    costList(i,:) = [Inf Inf];
end
costList(startNode,:) = [0 0];
foreach node do
    currentNode = foundList(nrFoundNodes);
    if currentNode = goalNode then
        Break
    end
    for toNode = 1 to nrNodes do
        if Adj_xy(currentNode, toNode) ≠ 0 then
            if costList(toNode,2) > (costList(currentNode,2)+Adj_xy(currentNode, toNode)) then
                costList(toNode,2) = (costList(currentNode,2)+Adj_xy(currentNode, toNode));
                costList(toNode,1) = currentNode;
            end
        end
    end
    shortList = find(costList(:,2) >= (costList(currentNode,2) & costList(:,2) < Inf));
    for foundNode = 1 to nrFoundNodes do
        shortList(shortList==foundList(foundNode)) = [];
    end
    minCostNode = shortList(1);
    minCost = costList(minCostNode,2);
    for j = 2 to length(shortList) do
        if costList(shortList(j),2) < minCost then
            minCostNode = shortList(j);
            minCost = costList(minCostNode,2);
        end
    end
    foundList = [foundList; minCostNode];
    nrFoundNodes = nrFoundNodes + 1;
end
c = costList(goalNode,2);
pathNode = goalNode;
sPath = [];
foreach node do
    sPath = [pathNode; sPath];
    if pathNode = startNode then
        if startNode = goalNode then
            sPath = [pathNode; sPath];
        end
        Return
    end
    pathNode = costList(pathNode,2);
end
```

Appendix B - Raycasting algorithm

In this appendix the raycasting algorithm as implemented in this research is shown in algorithm 3. This algorithm determines the visible waypoints from the *startpoint* and stores these in L_{visWp}. To determine whether a waypoint is visible a ray is cast from the start point to that waypoint. Then this line segment is checked for intersection against the line segments of each rectangular wind turbine safety boxes.

Algorithm 3: Raycasting

Result: List of visible waypoints for start point (L_{visWp})
foreach *waypoint* **do**
  intersect = False;
  if $waypoint \neq startpoint$ **then**
    foreach *safety box xy-edge* **do**
      (x_s, y_s) = *startPoint*; (x_w, y_w) = *waypoint*;
      (x_1, y_1) = *first safety box edge point*;
      (x_2, y_2) = *second safety box edge point*;

$$\text{dt1} = \begin{vmatrix} 1 & 1 & 1 \\ x_s & x_c & x_1 \\ y_s & y_c & y_1 \end{vmatrix} * \begin{vmatrix} 1 & 1 & 1 \\ x_s & x_c & x_2 \\ y_s & y_c & y_2 \end{vmatrix};$$

$$\text{dt2} = \begin{vmatrix} 1 & 1 & 1 \\ x_s & x_1 & x_2 \\ y_s & y_1 & y_2 \end{vmatrix} * \begin{vmatrix} 1 & 1 & 1 \\ x_c & x_1 & x_2 \\ y_c & y_1 & y_2 \end{vmatrix};$$

      if *dt* $\leq$ *0 and dt2* $\leq$ *0* **then**
        intersect = True; Break;
      end
    end
    if *intersect* = *False* **then**
      Add waypoint to L_{visWp};
    end
  end
end

References

[1] Jordan, S., Moore, J., Hovet, S., Box, J., Perry, J., Kirsche, K., Lewis, D., Tsz, Z., and Tse, H., "State-of-the-art technologies for UAV inspections," *IET Radar, Sonar & Navigation*, 2017. https://doi.org/10.1049/iet-rsn.2017.0251, URL www.ietdl.org.

[2] Pierce, S. G., Burnham, K. C., Zhang, D., Mcdonald, L., Macleod, C. N., Dobie, G., Summan, R., and Mcmahon D, "Quantitative inspection of wind turbine blades using UAV deployed photogrammetry," Tech. rep., uropean Workshop on Structural Health Monitoring, Glasgow, 2018. URL http://www.ndt.net/?id=23360.

[3] Stokkeland, M., Klausen, K., and Johansen, T. A., "Autonomous visual navigation of Unmanned Aerial Vehicle for wind turbine inspection," *2015 International Conference on Unmanned Aircraft Systems, ICUAS 2015*, 2015, pp. 998–1007. https://doi.org/10.1109/ICUAS.2015.7152389.

[4] Hoglund, S., "Autonomous Inspection of Wind Turbines and Buildings using an UAV," Ph.D. thesis, Norwegian University of Science and Technology - Trondheim, 2014. https://doi.org/http://hdl.handle.net/11250/261286, URL http://hdl.handle.net/11250/261287.

[5] Schäfer, B. E., Picchi, D., Engelhardt, T., and Abel, D., "Multicopter unmanned aerial vehicle for automated inspection of wind turbines," *24th Mediterranean Conference on Control and Automation, MED 2016*, 2016, pp. 244–249. https://doi.org/10.1109/MED.2016.7536055.

[6] Lai, S., Wang, K., and Chen, B. M., "Dynamically feasible trajectory generation method for quadrotor unmanned vehicles with state constraints," *Chinese Control Conference, CCC*, 2017, pp. 6252–6257. https://doi.org/10.23919/ChiCC.2017.8028351.

[7] Hehn, M., and Dandrea, R., "Real-Time Trajectory Generation for Quadrocopters," *IEEE Transactions on Robotics*, Vol. 31, No. 4, 2015, pp. 877–892. https://doi.org/10.1109/TRO.2015.2432611.

[8] Mueller, M. W., Hehn, M., and D'Andrea, R., "A computationally efficient algorithm for state-to-state quadrocopter trajectory generation and feasibility verification," *IEEE International Conference on Intelligent Robots and Systems*, 2013, pp. 3480–3486. https://doi.org/10.1109/IROS.2013.6696852.

[9] Stryk, O. Y., and Bulirsch, R., "Direct and indirect methods for trajectory optimization," Tech. rep., Mathematisches Institut, Technische Universitat Munchen, 1992. URL https://link.springer.com/content/pdf/10.1007%2FBF02071065.pdf.

[10] Mellinger, D., and Kumar, V., "Minimum snap trajectory generation and control for quadrotors," *Proceedings - IEEE International Conference on Robotics and Automation*, 2011, pp. 2520–2525. https://doi.org/10.1109/ICRA.2011.5980409.

[11] Velez, P., Certad, N., and Ruiz, E., "Trajectory Generation and Tracking Using the AR.Drone 2.0 Quadcopter UAV," *Proceedings - 12th LARS Latin American Robotics Symposium and 3rd SBR Brazilian Robotics Symposium, LARS-SBR 2015 - Part of the Robotics Conferences 2015*, IEEE, 2016, pp. 73–78. https://doi.org/10.1109/LARS-SBR.2015.33, URL http://ieeexplore.ieee.org/document/7402144/.

[12] Rousseau, G., Maniu, C. S., Tebbani, S., Babel, M., and Martin, N., "Quadcopter-performed cinematographic flight plans using minimum jerk trajectories and predictive camera control," *2018 European Control Conference, ECC 2018*, IEEE, 2018, pp. 2897–2903. https://doi.org/10.23919/ECC.2018.8550309.

[13] Bry, A., Richter, C., Bachrach, A., and Roy, N., "Aggressive flight of fixed-wing and quadrotor aircraft in dense indoor environments," *The International Journal of Robotics Research*, Vol. 34, No. 7, 2015, pp. 969–1002. https://doi.org/10.1177/0278364914558129, URL https://journals.sagepub.com/doi/pdf/10.1177/0278364914558129.

[14] Bouktir, Y., Haddad, M., and Chettibi, T., "Trajectory planning for a quadrotor helicopter," *2008 Mediterranean Conference on Control and Automation - Conference Proceedings, MED'08*, IEEE, 2008, pp. 1258–1263. https://doi.org/10.1109/MED.2008.4602025, URL http://ieeexplore.ieee.org/document/4602025/.

[15] Xu, H., Xu, X., Li, Y., Zhu, X., Jia, L., and Shi, D., "Trajectory planning of Unmanned Aerial Vehicle based on A* algorithm," *4th Annual IEEE International Conference on Cyber Technology in Automation, Control and Intelligent Systems, IEEE-CYBER 2014*, IEEE, 2014, pp. 463–468. https://doi.org/10.1109/CYBER.2014.6917508, URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6917508.

[16] Noreen, I., Khan, A., and Habib, Z., "A Comparison of RRT, RRT* and RRT*-Smart Path Planning Algorithms," Tech. Rep. 10, 2016.

[17] Anderson, S., "Comparing Offshore and Onshore Wind," Tech. rep., The Economics of Oil and Energy, 2013. URL http://pages.hmc.edu/evans/andersonwind.pdf.

[18] Vermeer, L. J., Sˆrensen, J. N., and Crespo, A., "Wind turbine wake aerodynamics," *Progress in Aerospace Sciences*, Vol. 39, 2003, pp. 467–510. https://doi.org/10.1016/S0376-0421(03)00078-2, URL https://www.sciencedirect.com/science/article/pii/S0376042103000782.

[19] Guerrero, J. A., Escareno, J. A., and Bestaoui, Y., "Quad-rotor MAV trajectory planning in wind fields," *Proceedings - IEEE International Conference on Robotics and Automation*, 2013, pp. 778–783. https://doi.org/10.1109/ICRA.2013.6630661.

[20] Stepanyan, V., and Krishnakumar, K., "Estimation, Navigation and Control of Multi-Rotor Drones in an Urban Wind Field," *AIAA Information Systems-AIAA Infotech @ Aerospace*, 2017. https://doi.org/10.2514/6.2017-0670, URL http://arc.aiaa.org.

[21] Dijkstra, E. W., "A Note on Two Problems in Connexion with Graphs," *Numerische Mathematlk l*, 1959, pp. 269–296.

[22] Broumi, S., Bakal, A., Talea, M., Smarandache, F., and Vladareanu, L., "Applying Dijkstra algorithm for solving neutrosophic shortest path problem," *International Conference on Advanced Mechatronic Systems, ICAMechS*, Vol. 0, 2016, pp. 412–416. https://doi.org/10.1109/ICAMechS.2016.7813483.

[23] Wang, H., Mao, W., and Eriksson, L., "A Three-Dimensional Dijkstra's algorithm for multi-objective ship voyage optimization," *Ocean Engineering*, Vol. 186, 2019, p. 106131. https://doi.org/10.1016/j.oceaneng.2019.106131.

[24] Bastankhah, M., and Porté-Agel, F., "A new analytical model for wind-turbine wakes," *Renewable Energy*, Vol. 70, 2014, pp. 116–123. https://doi.org/10.1016/j.renene.2014.01.002, URL https://linkinghub.elsevier.com/retrieve/pii/S0960148114000317.

[25] Sun, S., Sijbers, L., Wang, X., and De Visser, C., "High-Speed Flight of Quadrotor Despite Loss of Single Rotor," *IEEE Robotics and Automation Letters*, Vol. 3, No. 4, 2018, pp. 3201–3207. https://doi.org/10.1109/LRA.2018.2851028.

4

Conclusion and recommendation

The algorithm as presented in chapter 3 is a viable solution as a basis for trajectory generation. The objective of this research is to create a safe and efficient solution to inspect wind turbines by creating an optimal 3D trajectory generator for automated wind turbine inspector drones. Dijkstra's algorithm allows for taking into account the wind field, hence giving a more optimal solution in comparison existing solutions. In addition the proposed solution uses the in advance known wind turbine location and attitude hence creating a more optimal solution in general. The results in chapter 3 also shows that the algorithm can create safe routes back to the home waypoint, proving that the algorithm can handle an emergency.

However the results show a large offset due to the limitations of the controller. When using the generated trajectories as input to the controller, it generates velocity and acceleration references that are too large for the drone to follow, causing it to crash. To overcome this each link that exceeds a certain length is subdivided in smaller links, since limiting the velocity and acceleration references did not solve the problem. This ensures that the drone never receives a too high reference input, however it also significantly decreases the average speed of the drone and the accuracy of the results. It is recommended that this controller issue is addressed to generate more reliable results. Closely related to this issue is that the controller causes the drone to decelerates at every waypoint, even though this is not always necessary. This can partially be solved by addressing the controller, however it can be beneficial to create a motion profile after the 3D trajectory is generated, such as for path re-parametrization solution[1]. Lastly, related to the controller, the issue remains that the drone crashes with about 2 meters per second headwind. Is is recommended that the cause of this problem is found and in case this can have an influence on the results, all tests should be redone.

To increases the certainty that the provided algorithm is an improvement due to the wind optimisation, it is recommended to test with more realistic wind scenarios. The general wind field can be improved by using real wind data instead of a constant wind field with a uniformly distributed random influence. The reliability of the results can be increased if the actual path cost (travel time) without wind optimisation can be compared to the actual path cost with wind optimisation. Currently only the estimated path costs are compared to determine the increase in efficiency. This can be done by running the experiments again with the same variables however let the optimisation determine the trajectories without taking into account wind, resulting in the actual travel times.

To improve the proposed solution and achieve better results it is recommended to implement a model which estimates the wind field, including a wind turbine wake and wind shear model. This allows for better height optimisation, both during emergency and nominal trajectory generation. Secondly it is recommended to expand the graph which is used for optimisation to include more nodes than only the safety box corners. This allows for better horizontal optimisation and makes the solution less dependent on the wind turbine park lay-out. However a too large graph could impact the suitability for online use, hence the impact of graph size should be considered. Another improvement is to use energy consumption as a metric for the optimization instead of travel time [2], however due to the relatively long distances in between wind turbines this will probably only yield better results for manoeuvring close to the wind turbine. Hence this is only recommended for nominal operation and not for emergency procedures. Additionally to improve nominal path planning the Travelling Salesman Problem (TSP) which is implemented to order the turbines, can be expanded to a Multiple Salesman TSP to allow for inspection with multiple drones.

Bibliography

[1] Y. Bouktir, M. Haddad, and T. Chettibi. Trajectory planning for a quadrotor helicopter. In *2008 Mediterranean Conference on Control and Automation - Conference Proceedings, MED'08*, pages 1258–1263. IEEE, 6 2008. ISBN 9781424425051. doi: 10.1109/MED.2008.4602025. URL http://ieeexplore.ieee.org/document/4602025/.

[2] Carmelo Di Franco, Giorgio Buttazzo, C Di Franco, and G Buttazzo. Coverage Path Planning for UAVs Photogrammetry with Energy and Resolution Constraints. *Journal of Intelligent & Robotic Systems*, 83: 445–462, 2016. doi: 10.1007/s10846-016-0348-x. URL https://link.springer.com/content/pdf/10.1007%2Fs10846-016-0348-x.pdf.

[3] Sondre Hoglund. *Autonomous Inspection of Wind Turbines and Buildings using an UAV*. PhD thesis, Norwegian University of Science and Technology - Trondheim, 2014. URL http://hdl.handle.net/11250/261287.

[4] Sophie Jordan, Julian Moore, Sierra Hovet, John Box, Jason Perry, Kevin Kirsche, Dexter Lewis, Zion Tsz, and Ho Tse. State-of-the-art technologies for UAV inspections. *IET Radar, Sonar & Navigation*, 2017. ISSN 1751-8784. doi: 10.1049/iet-rsn.2017.0251. URL www.ietdl.org.

[5] S G Pierce, K C Burnham, D Zhang, L Mcdonald, C N Macleod, G Dobie, R Summan, and Mcmahon D. Quantitative inspection of wind turbine blades using UAV deployed photogrammetry. Technical report, uropean Workshop on Structural Health Monitoring, Glasgow, 2018. URL http://www.ndt.net/?id=23360.

[6] Björn E. Schäfer, Davide Picchi, Thomas Engelhardt, and Dirk Abel. Multicopter unmanned aerial vehicle for automated inspection of wind turbines. *24th Mediterranean Conference on Control and Automation, MED 2016*, pages 244–249, 2016. doi: 10.1109/MED.2016.7536055.

[7] Martin Stokkeland, Kristian Klausen, and Tor A. Johansen. Autonomous visual navigation of Unmanned Aerial Vehicle for wind turbine inspection. *2015 International Conference on Unmanned Aircraft Systems, ICUAS 2015*, pages 998–1007, 2015. doi: 10.1109/ICUAS.2015.7152389.