Master thesis

Master of Science in Space Engineering

School of Industrial and Information Engineering

Title

Author:
Alejandro de Miguel
Mat.number:
918897

Supervisor:
Dr. Mauro Massari
Co-Supervisor
Michele Maestrini

21st July 2021

POLITECNICO
MILANO 1863

To confine our attention to terrestrial matters
would be to limit the human spirit.
-Stephen Hawking

add a copyright page???

Abstract

THIS master thesis presents a sampling-based receding-horizon inspection algorithm that will be fed into a Neural Network in order to train it to solve on-orbit inspection optimal trajectories. The main goal of the thesis is to be able to demonstrate the Machine Learning—more precisely, Artificial Neural Networks—enhancement capabilities and advantages in terms of computational speed, which could potentially enable a reliable, autonomous, real-time onboard inspection algorithm.

Keywords:

Sommario

Parole chiave: Intelligenza Artificiale; Machine Learning; Reti neuronali; Deep Learning; Guida e Controllo;

translate: on-orbit inspections, heuristic methods, Delaunay triangulation,

Acknowledgments

List of Abbreviations

check that none are missing or they are not needed

Abbreviation	Meaning	Page or Section
OOI	On-Orbit Inspection	introduction
GNC	Guidance, Navigation and Control	introduction
OCP	Optimal Control Problem	introduction
APF	Artificial Potential Function	introduction
MPC	Model Predictive Control	introduction
ML	Machine Learning	introduction
AI	Artificial Intelligence	introduction
ANN	Artificial Neural Networks	introduction
NMT	Natural Motion Trajectories	introduction
RRT	Rapidly Exploring Random Trees	introduction
SBMPO	Sampling Based Model Predictive Optimisation	introduction
SBMPC	Sampling Based Model Predictive Control	inspection
FOV	Field of View	inspection
NMPC	Nonlinear Model Predictive Control	inspection

Contents

Chapter 1

Introduction

"This is the cite"
-Alejandro de Miguel

On-orbit inspections (OOIs) refer to a specific type of mission in which a spacecraft—called chaser or deputy—orbits within the vicinity of another spacecraft or body—called target or chief—with the goal of observing and inspecting it, by means of some inspecting instruments. The nature and requirements of the mission depend of many variables, but the most influential one is the target's ability to cooperate—which means that it would be able to control its own trajectory or attitude—. For non-cooperative, uncontrolled tumbling spacecrafts or bodies, OOIs become one of the most challenging space missions, as it will be the case for this thesis. To complete a mission of this kind, it is necessary to compute a trajectory that brings the chaser spacecraft as close to the target as it is needed—depending on the inspection instrument constraints—and under the right conditions; prioritise the safety of both the chaser and the target; and execute it in the minimal time possible and with minimal fuel consumption. These are the main objectives of the inspection guidance and control.

It is convenient to start by properly defining some important concepts. Guidance is the process of planning and determining a desired trajectory, in terms of both translation and rotation of the spacecraft. Of course this involves the computation of control forces and torques so that the trajectory can be executed properly. On the other hand, control—more precisely, feedback control—keeps the spacecraft close to the given trajectory. This is done by receiving navigation updates to guide the control actions in presence of disturbances, instrument noise and model uncertainties. The sum of these actions in the context of two close orbiting bodies, is referred to as relative GN&C (relative guidance, navigation and control). It constitutes

one of the pillars of on-orbit inspections, as the relative motion between two close bodies dominates in these types of scenarios.

Today, OOIs are mostly designed from ground and then sent to the inspector. This, in contrast with fully automated, real-time OOIs has several disadvantages that will be later discussed. Especially, due to the increasing number of satellites in orbit, the automatisation processes in space is becoming more and more necessary in order to keep a sustainable growth.

This is where machine learning (ML) comes in. Traditionally, OOIs algorithms have been computationally expensive, since they have to find the optimal solution for a complex problem with many degrees of freedom in a very limited time—and with the surge of small satellites and CubeSats, also with a very limited computational capability—. The introduction of machine learning, more specifically artificial neural networks (ANNs), will reduce the computational burden by taking advantage of their ability to learn the underlying basis of complex problems and speed the computations up, so that automated, real-time inspections algorithm can be successfully implemented in the space environment, with all the advantages that this implies.

narrate how the thesis will be structured

1.1 Motivation.

The goal of this thesis is to provide an efficient, autonomous and real-time on-orbit inspection method by mixing together the best state of the art inspection methods and the low computational costs of machine learning. Autonomous relative guidance represents a key technology enabler for the future of space industry. Multitude of different space disciplines, such as planetary entry, descent and landing (EDL); close proximity operations (*rendezvous* and docking, on-orbit refuelling, space debris removal and sample returns); scientific exploration (mapping and analysis of celestial bodies or debris) or autonomous inspection and servicing (AIS), depend on this technology.

In fact, NASA's Office of the Chief Technologist and the National Research Council published a report entitled "Restoring NASA's Technological Edge and Paving the Way for a New Era in Space" with a set of vital technologies to address the needs for the next generation of space programs. In the chapter TA04 "Robotics, Tele-Robotics, and Autonomous Systems"[1], one of the high-priority technologies is "Relative Guidance Algorithms", the basis of on-orbit inspections.

Despite its importance and high-priority, real-time autonomous relative guidance systems have not yet been developed or they are in the very early stages of research and testing. The guidance problem can be represented as an optimal control problem (OCP) that must be solved numerically, and the solution algorithms to this problem should be:

- Robust: if a feasible solution exists, it should be optimal.
- Real-time: the algorithms should be executed in a reasonable amount of time. This reasonable time will depend on the mission nature and computation capabilities.
- Verifiable: must be possible to analyse the performance and robustness of the algorithm.

in order to achieve a good degree of autonomy [2]. The autonomy capability for a spacecraft, especially when operating close to other objects or bodies, is essential for the success of future missions. Very often in space, remote control is not possible due to the delay of signals, which have to travel long distances (e.g a 26 minute delay to Mars). Also, with the recent increase in space commercialisation and consequently, the increase of objects in space, manual ground guidance and control will become difficult, unpractical and expensive, which makes this methodology untenable and calls for a more independent, automatised procedure. In addition, autonomy will directly increase the robustness and reliability of missions by reducing risks and human errors.

Another trend that supports autonomy and real-time capabilities is the usage of CubeSat and nanosats. This type of satellites, which have gained popularity over the last few years, are not equipped with big computation systems due to their obvious size and weight limitations. Therefore, they will also get directly benefited from low computational cost algorithms. This is where machine learning comes into play.

Machine learning is an application of Artificial Intelligence (AI). It uses algorithms based on statistics to find patterns in (usually) enormous amounts of data (numbers, images, words...).

Machine learning is an extensive field, it comprehends many different techniques and disciplines such as: Nearest-Neighbour Classifiers, Artificial Neural Networks (ANN) or Decision Trees or Evolutionary Algorithms (EA) among others. A brief discussion of the most relevant ones for this thesis is presented: briefly explain

- Decision Trees:

- Evolutionary Algorithms:
- Artificial Neural Networks:

1.2 State of the Art.

As it was briefly introduced before, the main challenge that autonomous spacecrafts face is solving the guidance and control problem with accurate dynamics and a constrained computational time.

Some successful techniques capable of dealing with autonomous spacecraft manoeuvring are:

- Apollo Guidance: briefly describe or delete it comprehends the guidance techniques developed by NASA during the Apollo Program. These classical techniques served as the basis for other standardised modern guidance techniques.
- Artificial Potential Functions (APF): a collision-free trajectory is created by guiding the spacecraft according to "forces", result of the gradient of potential functions. This functions are designed to be "attractive" towards the goal and "repulsive" to the obstacles. A limitation of this method is getting trapped in local minima (there are APFs with no local minima called navigation functions, but computing them is as hard as solving the whole planning problem).
- Model Predictive Control (MPC): it is a feedback law based on the iterated optimal solution. It uses a dynamic model f and the current state of the spacecraft as initial condition. Thus, it optimises the propagated or predicted state response over a finite-time horizon. However, when solved, only the initial segment is actually used, after which the process is iterated again until it converges to the goal. Because of this renewal over an updated horizon the MPC is also called *receding horizon* or *moving horizon* optimal control. Some important advantages of this method is its capability to handle, time pointwise, states and constraints; to withstand time delays; and to reconfigure in the presence of failure modes.
- Mixed-Integer Linear Program (MILP): this solvers are used for simple cases, when treating with linear dynamics and discrete decision variables, often together with a MPC to include simple logical constraints (e.g mode switching and collision avoidance).extend explanation

- Motion Planning Algorithms: they generate decision sequences to safely guide a vehicle from an initial state to a target state (or goal). It is a well studied type of algorithms, so its framework is general enough to be applied to different types of vehicles (robots, rovers and spacecrafts)

Each technique has its advantages and disadvantages. When dealing with time-varying constraints (debris, other spacecrafts), logical modes (e.g. safety modes) and real-time operations (i.e due to time limitations, tradeoffs between feasibility and optimality have to be taken) the previous techniques, except motion planning, fall short. In contrast, motion planning has proven to be an effective solver for complex real-time kinodynamic problems, such as the proximity operations one. In addition, it can be used in many scenarios due to its geometric modelling independence and the possibility of implementing differential constraints and robustness verifications. double check this paragraph Motion planning techniques can be classified into two categories:

- Exact or combinatorial: they represent the portion of the configuration space occupied by obstacles. This guarantees a solution, if any exists. Due to computational issues this approach is applied to low-dimensionality, static environments—not really the case of the thesis—.

- Approximate or sampling-based: they explore possible paths by sampling the configuration space. A collision detection algorithm, which can be designed independently, determines the safety of the manoeuvre. This allows motion planning to be formulated for any particular model. The obvious disadvantage is that the existence of solutions cannot be guaranteed for a finite time.

Due to the requirements of the project, sampling-based methods are the best fit explain better this choice over others (e.g RRT). Proximity operations are, once again, characterised by complex dynamics and constraints that this kind of planning algorithms can handle autonomously.

Despite this, not much work in spacecraft control systems has been done following this approach, as it is usually related to feasible plans, not optimal ones, which is particularly important on spacecrafts and space missions. However, algorithms of this kind that offer some optimality (usually weak) are currently receiving a lot of attention.
add machine learning state of the art

Flight demonstrations, challenges and future work.

FLIGHT DEMONSTRATIONS.
As mentioned before, spacecraft autonomy needs of robust and optimal control techniques. Even though some progress in autonomous systems has been made (e.g Mars Curiosity), they are too expensive and too mission-specific to be implemented in a general, reliable way in other missions or scenarios. On top of this, many of the existing methods cannot be considered as fully mature, as some anomalies (or even actual collisions such as the satellite DART against MUBLCOM satellite [3]) in previous test flights and mission demonstrations (see [4], [5], [6])suggest. This puts in perspective the degree of difficulty of these real-time autonomous decision-making systems.

FUTURE WORK.
The work in [7] proposed an extended linearisation technique, the State-Dependent Riccati Equation (SDRE), as a new suitable control law for relative guidance system. Simulation were carried out using real mission data and including space perturbations to understand its applicability to real situations. It takes advantage of the Natural Motion Trajectories (NMT) to orbit around the target and inspect it. However, it does not propose any inspection strategy—such as what features of the target are observed, during how much time or under what illumination conditions—, nor includes safety.

In [8] two motion planning algorithms are presented, namely Rapidly Exploring Random Trees (RRT) and Sampling Based Model Predictive Optimisation (SBMPO), for two different guidance problems (landing on a small body by RRT and its observation by SBMPO). It considers two real mission scenarios to test the algorithms capabilities, the landing of *Rosetta* and the observation of *Didymain*. The validation of these methods is done by comparing them with classical direct optimisation techniques (e.g direct collocation, multiple shooting) obtaining that, despite the local optimality of the direct techniques, the best solutions of the planning algorithms were more than comparable to the ones from the classic methods, while avoiding the large computational times and parametrisation problems of the last. They also fulfilled the "high-level" mission objectives (i.e a task that cannot be directly translated into a predefined trajectory) Lastly, SBMPO included safety constraints check if RRT also can include safety constraints and an observation model (icosahedron) to make sure all faces were observed at least a minimum time $T_{obs,min}$.

1.3 Selected Inspection algorithm and Artificial Neural Network type.

As it was previously seen, state-of-the-art techniques for autonomous close operations include Apollo guidance, Artificial Potential Functions and Model Predictive Control. However, these techniques, although ideal for static uncluttered environments, are not enough whenever the priority relies on the optimisation, logical modes (e.g. safety modes) and time-varying constraints. In these contexts, a technique originally developed for robotics, named motion planning technique, has arisen as a promising alternative. Unfortunately, it is still in the process of being adapted and proven for spaceflight. The motion planning techniques rely, at the same time on algorithms like the Sampling-Based Motion Planning to reduce the computational burden that arises from the complexity of the problem. SBMP algorithms rely on sampling the input space to then construct feasible paths. The idea of sampling the input space and then propagating the feasible commands is a powerful tool—specially to high-level guidance objectives as in this thesis case—that has been successfully proven in other works ([9], [10]).

In this thesis this type of approach will be followed. More precisely, the inspection algorithm will be based on a receding-horizon trajectory planning algorithm adapted to passively safe on-orbit inspections [11]. It is an algorithm based on classic SBMPs, but the trajectory propagation is made in a receding-horizon manner, rather than the classic search tree.

The details of the proposed algorithm will be explained later on in Chapter 3

add the selection argument for the ANN

maybe some of the introduction can be here described

Chapter 2

Relative Guidance and Control

"This is the cite"
-Alejandro de Miguel

2.1 Introduction.

It is useful to briefly define a few concepts that are key in optimal control and trajectory planning. A *state* is the set of information that completely defines, mathematically, the motion of a system. It is usually represented as a vector $\boldsymbol{x} \in \mathbb{R}^d$ which can include positions, velocities, masses or other physical variables. A *control* is also a set of mathematical variables, but this time of the variables one can have control over. In other words, they can be modified to produce a given change in the state. It is usually represented as a vector $\boldsymbol{u} \in \mathbb{R}^{N_u}$. They are the inputs of the system, such as actuators, thrust forces or control torques. Lastly, an *objective*, often represented as J, is a measure of the performance of the system. It can be also viewed as the output of the system.

2.2 The Optimal Guidance Problem.

write the optimal control problem

2.3 Literature Review.

2.4 Equations of motion.

Relative motion dynamics

talk about the different equations describing the relative motion problem: HCW, LERM, NERM... nonlinearities, coupling of attitude and translation....

Target attitude motion.

2.5 Integration of equations

mention something about the integrator and the fact that we interpolate in order to be less computationally expensive?

Chapter 3

On-Orbit Inspection

"This is the cite"
-Alejandro de Miguel

AMONG the most complex space mission types, such as *rendezvous*, debris removal, servicing and refuelling missions, on-orbit inspection could be considered as one of the most challenging ones, especially when involving a non-cooperative, tumbling target. The complexity of this type of missions is related to the high-level objective they propose and the high number of constraints implicit in the success of the mission (e.g. collision avoidance and safety, target's proper illumination conditions, observation instrument constraints, chaser's attitude and distance relative to the target...). The motivation for developing good on-orbit inspection algorithms, trajectories and strategies comes from the powerful benefits and opportunities that inspections yield, being the key enabler for carrying out effectively other types of missions which require of information that can only be extracted from an on-orbit inspection. The proposed inspection algorithm that will be explained is based on the work of [11]

3.1 The Inspection Problem

The inspection problem could be briefed as the problem of finding the optimal guidance and attitude control (most commonly in terms of time and fuel) that allows a chaser spacecraft to perform a proper inspection—which depends on many variables such as illumination, safety or the inspection equipment constraints (e.g. the field of view, resolution distance, the viewing orientation...)—by taking images or other measures of a target spacecraft.

The applications that come from a good inspection are numerous. In many scenarios, it is indispensable that a target (either a vehicle or another orbiting object) is properly inspected by a chaser before being able to execute a further proximity operation, especially when the target is unknown, non-cooperative and/or tumbling uncontrollably in space. Just to name some fields that are directly benefited of inspections:

- Active space debris removal: the number of space debris orbiting Earth is growing fast due to the increasing number of launches per year and the later on-orbit collisions, which just make the number of space debris grow exponentially. It is an imminent problem that needs to be tackled as soon as possible. It constitutes one of the main challenges the space industry has to face [12].

- Unmanned on-orbit servicing (OOS): some initiatives have been successful, and it is an active investigated field due to its commercial appeal.

- Scientific interest missions: in many scientific missions the main goal is to observe (optically or by other means) a planet, moon, asteroid... which indirectly involves an inspection trajectory and planning. In addition to this, other types of mission in which the main goal can be different (e.g landing, sample collecting..), require of a previous inspection of a body in order to analyse its most interesting features, landing sites or atmosphere composition for atmospheric entry for example.

3.2 Problem Statement

As it was mentioned before, this thesis will study an on-orbit inspection of a non-cooperative, uncontrolled tumbling vehicle placed on a circular orbit around Earth. The inspection will be carried out by taking images with a passive camera installed on the chaser vehicle. The success of the mission will depend on: the ability to observe certain parts of the target during sufficient time and under proper conditions; the mission cost in terms of time and propellant; and last but not least, the safety of both the target and the chaser. To guarantee the last goal, all eligible trajectories must be passively safe (i.e their collision probability under anomalies or lack of spacecraft control must be smaller than a defined safety threshold, which must ensure that the trajectory is safe for a sufficient period of time so that ground response can be executed).

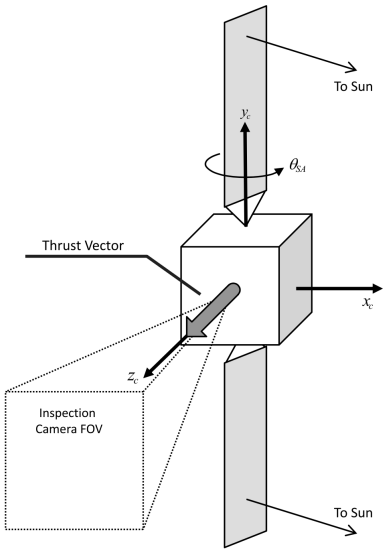

Figure 3.1: Chaser's body reference frame and configuration of the thruster, inspection camera and solar arrays.[11]

To further realistically model the inspection mission, vehicle and hardware constraints are considered. In fact, to really test the limits of the algorithm, a simple but highly constrained model will be implemented. Figure 3.1 depicts a sketch of the chaser vehicle, where the configuration of the thruster, camera and solar arrays can be visualised.

The chaser vehicle will be equipped with a single, non-orientable passive visible camera, which means that, in order to observe a certain feature of the target, the camera will depend on the relative attitude of the chaser with respect to the target, on the illumination conditions and on the eclipse periods. The thrust will be given by a single low-thrust electric engine that will perform the desired trajectory manoeuvres. The thrust vector is aligned with the camera boresight axis. Therefore, attitude manoeuvres are necessary to redirect the thrust vector or the camera axis, depending on the phase of the mission to be performed. These manoeuvres are performed by the reaction wheels-based attitude control system and thus, must comply with the maximum torque and momentum that the reaction wheels can exert. Lastly, due to power constraints, the engine cannot be used during eclipses and, in addition, whenever the spacecraft is outside the eclipse, solar arrays need to maximise solar flux. The solar array drive mechanism that enables its movement is constrained by a maximum rotation rate that cannot be exceeded. last constraints not implemented in code yet

3.3 Challenges and constraints

The goal for an inspection guidance is to bring the chaser spacecraft as close to the target as it is needed while consuming minimum fuel and ensuring safety. This requirements introduce complex, non-convex trajectory constraints into the optimal control problem. ref section if opt.control problem is finally explained.. However, they are not the only constraints that should be considered in an inspection mission. In the following, relevant examples of inspection constraints will be briefly summarised:

- Instrument Field of View (FOV): it is an important constraint that depends on the instrument capabilities. It constraints the distance range between chaser and target, and also the maximum angle at which a target feature can be observed from.

- Illumination Conditions: not only the target needs to be illuminated (not in eclipse), but properly illuminated. This constraint has been introduced by means of an angle β between the viewing direction of the feature and the Sun direction.

- Power Limitations:

- Actuator Constraints:

- Plume Impingement:

- Thruster Limitations:

- Collision Avoidance:

- Thruster Silence Times:

- Limited Propellant Use:

- Safety:

- Uncertainties

maybe they are too many, some of them wont be taken into account in this thesis

3.4 Observation Model

The main goal of the inspection is observing the target . Then, it is useful to find a way to quantify the mission completion percentage and to discretise the target area into discrete, observable, interesting features, from which the target surface could be, for example, reconstructed, or they could also be important scientific aspects that are desired to be observed specifically.

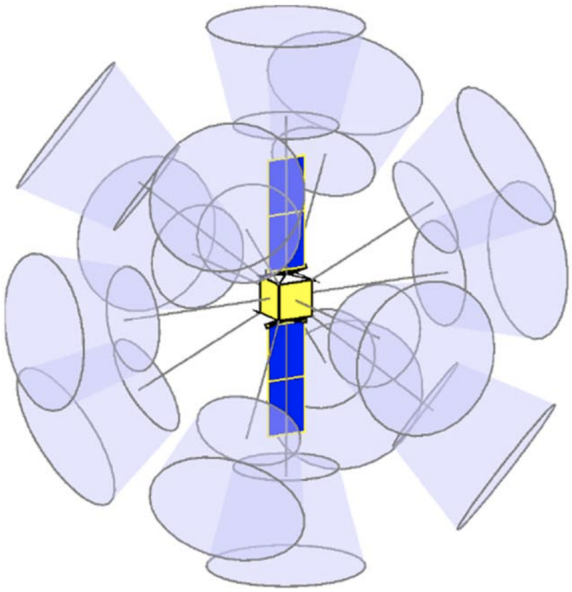

Figure 3.2: The lines point to the target observable features (faces and corners of the vehicle) while the cones represent the valid observation regions for each feature.[11]

Figure 3.2 depicts the selected observable features (the center point of each face and the vertices of the chaser vehicle) and the observation regions (cones) that allow a feature to be observed, taking into account the field of view (FOV), distance and illumination constraints. Each feature is unequivocally defined by a viewing direction, that is, a vector coming from the feature, normal to its surface.

The quality of the images is preserved if some conditions are fulfilled:

- The distance between the target and the chaser must be limited to some values: $r_{min}^{obs} \leq r \leq r_{max}^{obs}$
- The angle α, defined as the angle between the viewing direction of the feature and the relative position vector target-chaser, must be smaller

than a maximum threshold: $\alpha \leq \alpha_{max}$

- The angle β, defined as the angle between the viewing direction and the Sun direction, must be smaller than a maximum threshold: $\beta \leq \beta_{max}$
- The target must not be eclipsed by the Earth during the observation phase as it would not be illuminated during this period.

Now, the inspection mission progress can be quantified by a completion percentage M_c, given by Equation (3.1):

$$M_c(t) = \frac{100}{n_p \Delta T_{goal}} \sum_{i=1}^{n_p} min(\Delta T_{goal}, \Delta T_{obs,i}(t)) \tag{3.1}$$

where ΔT_{goal} is the cumulated time a feature has to be observed for to be considered fully observed; n_p is the number of observables features of the target (depending on the target angular motion, some features may not be observable); and $\Delta T_{obs,i}$ defines the time a feature has been observed for. It is easy to interpret that, when $\Delta T_{obs} \geq \Delta T_{goal}$ for every observable feature, the mission completion parameter $M_c = 100\%$, indicating that the mission has been completed.

3.5 Planning Algorithm

Mission design

Because of the complexity of the high-level inspection requirement, the problem of finding the optimal trajectory able to completely observe a target has been reduced to finding the sequence of optimal trajectories (or inspection legs) that completely observe a target. An inspection leg is, at the same time, divided into several phases or sequences of events that, ultimately, allow the chaser to observe the target. It is important to highlight here the receding-horizon behaviour of the algorithm. Each manoeuvre yields the optimal single inspection leg (computed in a finite time), and inspection legs are performed until the inspection mission is completed. Therefore, no a-priori inspection sequences are given to the algorithm, it is free to choose all the trajectories based on the score it assigns to each leg (the score function will be later discussed). Of course, in order to compute which feature can be observed at each time, it is assumed that the attitude of the target is known—by for example, propagating its attitude dynamics,

knowing its initial attitude conditions at the beginning of each leg, or using an onboard pose estimation filter— during the inspection leg duration.

However, it is worth to mention that differently to [11], which focus on computing the sequence of optimal inspection legs that fully observe a target vehicle, in this thesis the focus will be on computing the single inspection leg that maximises the number of observed features of a target vehicle. This distinction is due to the fact that when computing an optimal sequence of legs, the number of required legs to fully observe a target is neither known or constant. This is a major drawback when training neural networks, as they need a constant number of inputs in each training example to learn from. Therefore, having inspection leg sequences of different sizes (e.g. having a sequence with 3 legs, another one with 5 legs and so on) is not really an option. Therefore, the proposed inspection algorithm will be used as a guideline, but a new single optimal inspection leg algorithm will be developed in this thesis.

Inspection legs

An inspection leg comprehends different phases that will be discussed in the following. An inspection leg starts with 1) Manoeuvre computation, of duration ΔT_{comp} in which the chaser's computer onboard computes the trajectory (manoeuvre) that maximices the observation of the target; 2) Manoeuvre attitude acquisition, of duration ΔT_{att}, in which the chaser acquires the attitude that positions the thrust vector in a way that allows to perform the desired inspection trajectory; 3) Manoeuvre execution, of duration ΔT_{mant}, in which the engine propulses the chaser to follow the computed trajectory; 4) Target pointing acquisition, of duration ΔT_{att}, in which the chaser changes its attitude so that the passive camera can point towards the target; 5) Observation, of duration ΔT_{obs}, this is the part of the inspection leg in which the observation can be made. It is worth to point out that not all this period of time is actual useful feature observation. As it was explained before, the observation is tightly constrained to illumination conditions, relative distances or eclipse phases. Also, notice that the times of each phase can be considered constant, as they depend on the onboard computer and vehicle capabilities, except for ΔT_{man}, which depends on the Δv required to perform each inspection leg, and ΔT_{obs}, which depends on the total duration of the inspection leg.

Algorithm Routine

The proposed algorithm is a sampling-based model predictive optimisation (SBMPO) algorithm, inspired by the work of [10], [13], [14], [15], [16] and [11]. In consonance with the mission goal, the goal of the algorithm is finding the optimal inspection leg that maximise the observation time while minimising fuel consumption and time duration. This will be done by exploring the search space of all the admissible legs and selecting the optimal one. To do so, a leg will be unequivocally defined by its duration and the Δv needed to execute it: $\boldsymbol{s} = [\Delta \boldsymbol{v}^T, \Delta T_{leg}]^T \in \boldsymbol{S}$. Notice that an admissible leg will have to belong to the four-dimensional admissible search space $\boldsymbol{S} \subset \mathbb{R}^4$ defined as:

$$\boldsymbol{S} : \left\{ ||\Delta \boldsymbol{v}|| \in [\Delta \boldsymbol{v}_{min}, \Delta \boldsymbol{v}_{max}] \cup \{0\}, \Delta T_{leg} \in [\Delta T_{leg}^{min}, \Delta T_{leg}^{max}] \right\} \tag{3.2}$$

The question that arises at this point is how the search space exploration is carried out. It will be performed by a heuristic sampling, initialised randomly at the beginning to be later on guided based on the most promising zones (the ones with a better scoring of S).

The logic of the algorithm is the following: it is initialised by sampling n_{s0} uniform samples within S, which is analogous to sample points uniformly from a four-dimension sphere (as the legs are defined in four dimensions, three of them representing the Δv components, depicted in Figure 3.3, and the fourth being the inspection leg time). This process is called *initial mesh*. Then, each sampled leg is propagated using the correspondent dynamic equations (discussed in Chapter 2). The next step is scoring each propagated trajectory following the scoring criteria of the method. It also useful to discard not valid legs, as it gives a null score to those trajectories that are not feasible (due to being unsafe, or escape from the target vicinity, not fulfilling the attitude constraints...). The idea behind the scoring is to refine the search space based on the previous sampled trajectories. This will be done by the *refine mesh* method. Basically it is a heuristic function to bias the new samples towards the most optimal/interesting regions of the search space. This function will sample n_S additional samples and can be called several times. Finally, the leg with the highest score will be selected. In the original algorithm this process would be repeated until mission completion reaches 100% (the whole spacecraft is inspected), but under the circumstances of this thesis the goal is selecting just one optimal leg.

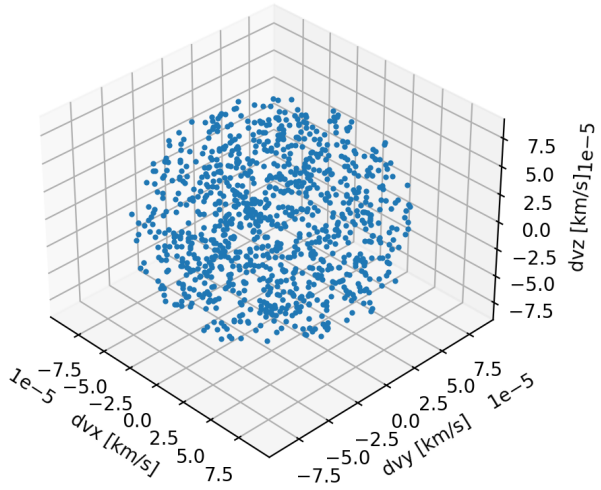

Figure 3.3: Δv sphere of outer radius Δv_{max} and inner radius Δv_{min} from which a thousand points have been uniformly sampled.

Scoring

An important advantage of this algorithm is that any score function can be implemented, so it can serve different planning problems or strategies. Depending on the goal, one or another strategy can be prioritised when choosing the optimal legs. It is also useful to penalise or completely remove those legs where mission conditions or constraints are violated by assigning them a null score—for example in the events of the chaser escaping the vicinity of the target, the chaser's angular motion exceeding the allowed reaction wheels and solar array limits, propulsion manoeuvres performed during eclipse or unsafe trajectories—. Equation (3.3) has been designed specifically in this thesis to compute the score of the sampled legs.

$$C = \frac{w_{\Delta v}}{w_{\Delta v} + \Delta v} \left[w_\gamma \int_{\Delta T_{leg}} f(t) \frac{\pi - \gamma(t)}{\pi} dt + \sum_{i=1}^{n_p} \Delta T_{useful,i} + m_i \right] \tag{3.3}$$

where w_γ and $w_{\Delta v}$ are weighting terms; f(t) is a weighting function to reward trajectories that maintain the distance to the target between the maximum and minimum limits, defined as:

$$f = \begin{cases} r/r_{obs}^{min} & \text{if} \quad r < r_{obs}^{min} \\ 1 & \text{if} \quad r_{obs}^{min} \leq r \leq r_{obs}^{max} \\ (r_{obs}^{max}/r)^3 & \text{if} \quad r > r_{obs}^{max} \end{cases} \tag{3.4}$$

$\gamma(t)$ is the angle between the position vector target-Sun and the relative position vector target-chaser; ΔT_{useful} is the useful observation time during which a feature can be properly inspected during an inspection leg (it is worth to mention that this time is capped at ΔT_{goal} as longer times than this can be considered useless as the feature has already been completely observed. Also, if a feature has not been observed for at least ΔT^{min}_{useful}, the useful time will be considered as null); and Δv is the magnitude of the manoeuvre required impulse. Notice that the multiplying term at the beginning of the equation weights the score in terms of propellant expenditure, favouring those legs requiring a low Δv. From the integral term it can be deduced that the geometry of the leg is evaluated, favouring those trajectories on the sunlit side of the target and respecting the distance limits. On the other hand, the summation term evaluates both the useful observation time the leg can provide for each feature and the number of features that are observable. The last consideration is quantified by m_i, a tuneable variable introduced to reward those trajectories able to observe more features. Whenever a leg is not able to inspect a feature ($\Delta T_{useful} = 0$), $m = 0$. If the legs allow the observation of a feature, m takes the designed value.

Heuristic Mesh Refinement

The `refineMesh` function is inspired in [15]. Its purpose is to heuristically analyse the existing samples and the search space, looking for interesting zones to sample new inspection legs from. To achieve this, it relies on a Delaunay triangulation, subdividing the search space into four dimensional simplices created by the initial available samples from `initMesh`. Each vertex of each simplex is a scored inspection leg $\boldsymbol{s}$. Then, each simplex is given a score, computed by Equation (3.5):

$$J_q = V_q^{\eta_V} \left(1 + \eta_M \frac{M_q}{M_{max}} + \eta_G \frac{G_q}{G_{max}}\right), \quad \forall q = 1, ..., N \tag{3.5}$$

where V_q is the volume of a simplex; M is the maximum trajectory score of the vertices of a simplex; G is the maximum trajectory score gradient of a simplex, M_{max} is the maximum score of all the vertices; and G_{max} is the maximum score gradient of the triangulation.

This cost is used to identify the regions that either have not been explored in depth—associated with large volumes, so it is given a high score—, that will yield a high mission score S or the ones that are characterised by high score gradients—potential to have interesting features or scores—. In this sense, the weighting coefficients η_V, η_G and η_M give a lot of flexibility

to the heuristic search, opening possibilities for different types of searches such as very explorative ones or very aggressive ones.

Then, the simplices are sorted in function of their scores so that new samples will be more likely drawn from the most successful simplices. This likelihood has been chosen to follow a weighted uniform distribution, so that even though all simplices can be selected, the probability to be selected depends proportionally to its score, (see [17] for more details). When the simplices to sample from are chosen, the new samples are randomly drawn within the volume of the simplices following the work of [18].

Safety and attitude constraints

write this subsection if/when implemented

3.6 Results

In this section the results of the inspection algorithm are shown. Table (make a table) gather all the initial conditions prescribed for the case taken as an example. Note that some of these initial conditions are randomly initialised (e.g the initial angular velocity of the target or the initial attitude of the target) in order to solve the inspection problem for different scenarios. Figure 3.4 may not be useful anymore since we are not doing leg sequences, just single legs depicts the result of a complete inspection ($M_c = 100$). The blue sphere, with radius r_{min} represents the target and its safe sphere, a region where the chaser should not enter. Each orbit colour represents a different inspection leg.

Now, Figure 3.5 shows the magnitude of the relative position vector target-chaser. It also includes some regions: the r_{max} region which limits the maximum distance threshold in order to not escape from the target; the r_{min} region which limits the minimum distance threshold in order to not impact with the target; and the observation zone, which limits the range that allows the chaser to observe the target. No observation should be possible out of this zone.

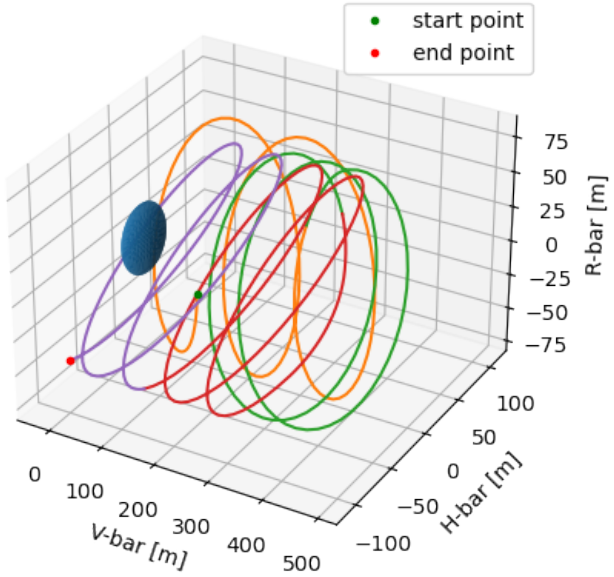

Figure 3.4: Optimal legs inspection sequence. Each orbit color represents a different inspection leg.

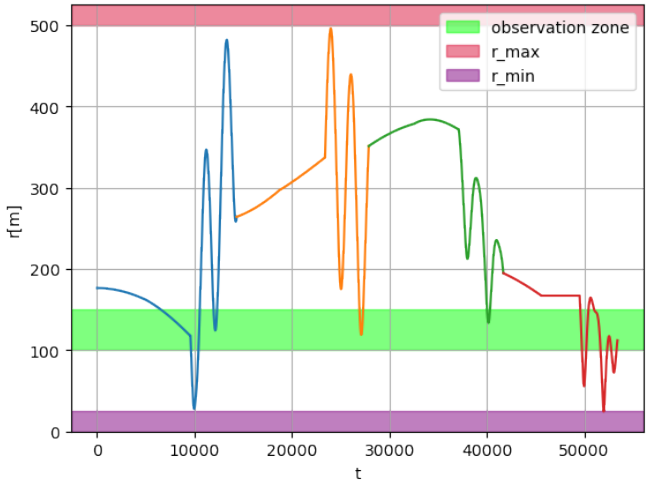

Figure 3.5: Distance [m] between the target and the chaser during the whole mission.

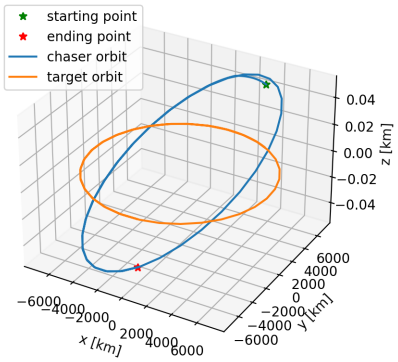

Figure 3.6: Target and chaser orbits around the Earth (ECI frame). The z-axis has been scaled to make evident the difference between the orbits. mention figure in the text somewhere

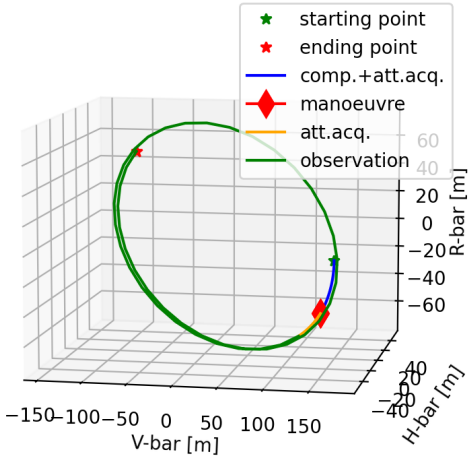

Figure 3.7: Inspection leg around the target. Depicted in the LVLH target frame. mention figure in the text somewhere and change colours by types of lines. Also set a standard size for the figures

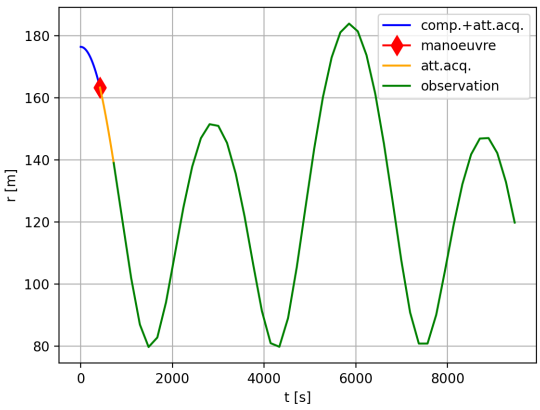

Figure 3.8: Distance [m] between the target and the chaser during an inspection leg. The different inspection leg phases are also highlighted. mention figure in the text somewhere and change colours by types of lines. Also set a standard size for the figures

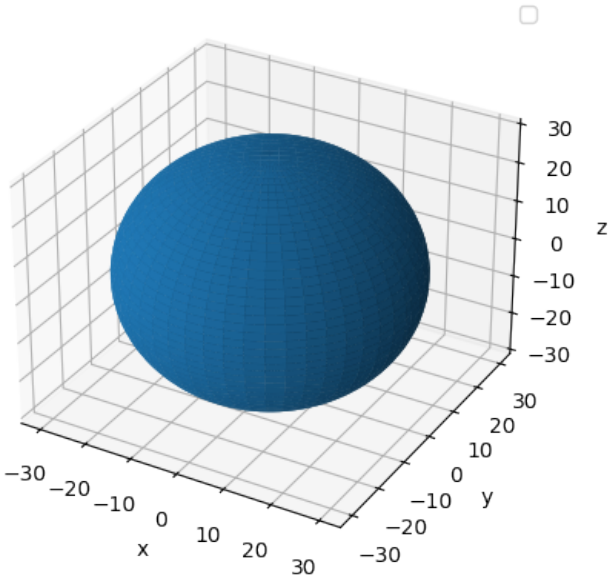

Figure 3.9: Example. Example, delete

Chapter 4

Machine Learning

"This is the cite"
-Alejandro de Miguel

dont know if this chapter is needed, the idea is to deeper describe machine learning, specially the chosen method, see briefly how it was applied before (maybe this better in introduction), how i design and parametrize the neural network

4.1 Introduction

brief introduction to machine learning and/or artificial intelligence and/or neural networks In machine learning three different distinctions can be made depending on the type of learning a neural network is presented:

- Supervised learning: the neural networks receives a—usually large—database with example inputs and desired outputs. The learning is carried out by comparing how well the neural network is able to approximate the desired outputs given the inputs. It is after many iterations that the neural network is able to adapt its parameters to properly map the hidden dynamics that the system represented by the set of inputs/outputs has.

- Unsupervised learning: it is similar to the supervised learning, but this time the database is given with no labels. This way, the neural network is forced to to look by itself for patterns in the data, to hopefully find a structure in the inputs.

- Reinforcement learning: the neural network actively interacts with an environment in order to achieve a goal or task. It is during this

interaction with the environment that the neural network learns about how to effectively achieve its goal by means of trial and error and feedback through "penalties" and "rewards".

As introduced in the previous sections, in this thesis the neural network will be trained in a supervised manner. Therefore, it is needed to provide the neural network with a database of examples from which it can extract the hidden dynamics and patterns of the on-orbit inspections.

4.2 Database Generation

In other machine learning applications, creating a dataset is not even necessary as there may be a large amount of data already available to use to this end. Unfortunately, this is not the case for this work and thus, creating a database has become one of the main tasks of the thesis, as the learning process and capabilities of the neural network will depend mostly on having a good database where it can learn from. Generating or creating a database is one of the most critical points for a successful neural network. Having a database too short and the neural network may not have enough examples to learn; too large and you will encounter overfitting problems as well as having wasted resources (creating elements of a database takes both computational power and time); include examples that are sparse or very different from one another and they may confuse the network, making more difficult the learning process; leave some important states or parameters of the system out of the database and the neural network may find impossible to learn the whole underlying dynamics of the system.

A priori there is not any theoretical rule or theorem about the minimum requirements a good database should have in order to be useful for the learning of a neural network—e.g. how large the database should be or how to measure the quality of the data—at best some rules of thumb, references from previous works and the classic trial and error. In fact, due to the high computational demand of the inspection algorithm and the limited computational resources available for the development of this thesis, the generation of a good database has become of paramount importance.

The main goal when generating the database is minimising computational time and power. In order to do so, the size of the database should be kept to a working minimum and the developed inspection algorithm as efficient and fast as possible without disregarding data quality and accuracy. Combining these two requirements is in fact a big challenge.

The database has been generated by randomly initialising the parameters that will be used as inputs to the neural network, and solving the in-

spection problem with the inspection algorithm presented along Chapter 3 in order to obtain the outputs for the neural network. This process has been repeated until a decent database size has been obtained.

In this thesis, the parameters defining the attitude of the target have been selected as inputs to the problem—namely three variables describing the tumbling angular velocity of the target and four variables describing its attitude, expressed in quaternions—. Therefore, the input of both the database and the neural network will be determined by 7 parameters: $input = \{\boldsymbol{\omega}_t, \boldsymbol{q}_t\}$. Regarding the output, recall from Section 3.5 that just four parameters are enough to fully define an inspection leg, and so: $output = \{\Delta\boldsymbol{v}, \Delta T_{leg}\}$.

As for the size of the database, typical numbers for neural network's supervised learning are database sizes in the range of the ten thousands?. In this case, a database of size write the database size has been enough to train the neural network. The computational time per leg for number of sampled legs leg samples is about execution time seconds using a $2.6GHz$ *Intel® Core i7* processor and running on *Python 3.8*, which makes up for a total computing time of time for database to generate the database.

4.3 Neural network architecture

grid search method paper: James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. Journal of Machine Learning Research, 13(Feb):281?305, 2012.

cyclical learning rates (Smith, 2017) and cyclical momentum: Leslie N Smith. Cyclical learning rates for training neural networks. In Applications of Computer Vision (WACV), 2017 IEEE Winter Conference on, pp. 464?472. IEEE, 2017.

Neural networks: tricks of the trade? (Orr & Muller, 2003) that contains several chapters with practical advice on hyper-parameters, such as Bengio (2012)

4.4 Results

Chapter 5

Results

"This is the cite"
-Alejandro de Miguel

Chapter 6

Summary, Conclusion and Future Work

"This is the cite"
-Alejandro de Miguel

List of Figures

List of Tables

Bibliography

[1] National Research Council. 'NASA Space Technology Roadmaps and Priorities: Restoring NASA's Technological Edge and Paving the Way for a New Era in Space'. In: The National Academies Press, 2012. Chap. Appendix G: TA04 Robotics, Tele-Robotics, and Autonomous Systems. DOI: 10.17226/13354.

[2] Joseph Starek et al. 'Spacecraft Autonomy Challenges for Next-Generation Space Missions'. In: *Lecture Notes in Control and Information Sciences* 460 (Sept. 2016). DOI: 10.1007/978-3-662-47694-9_1.

[3] NASA. 'Overview of the DART Mishap Investigation Results For Public Release'. In: 2006.

[4] Isao Kawano et al. 'Result and evaluation of autonomous rendezvous docking experiment of ETS-VII'. In: *Guidance, Navigation, and Control Conference and Exhibit*. DOI: 10.2514/6.1999-4073. eprint: https://arc.aiaa.org/doi/pdf/10.2514/6.1999-4073. URL: https://arc.aiaa.org/doi/abs/10.2514/6.1999-4073.

[5] Thomas Davis and David Melanson. 'XSS-10 microsatellite flight demonstration program results'. In: *Proceedings of SPIE - The International Society for Optical Engineering* 5419 (Aug. 2004). DOI: 10.1117/12.544316.

[6] R. T. Howard et al. 'Orbital Express Advanced Video Guidance Sensor'. In: *2008 IEEE Aerospace Conference*. 2008, pp. 1–10. DOI: 10.1109/AERO.2008.4526518.

[7] Franzini giovanni. 'Nonlinear Control of Relative Motion in Space Using Extend Linearization Technique'. PhD thesis. 2014.

[8] Thierry Simeon Francesco Capolupo and Jean-Claude Berges. 'Heuristic Guidance Techniques for the Exploration of Small Celestial Bodies'. In: *IFAC-PapersOnLine* 50.1 (2017). 20th IFAC World Congress, pp. 8279 –8284. ISSN: 2405-8963. DOI: https://doi.org/10.1016/

j.ifacol.2017.08.1401. URL: http://www.sciencedirect.com/science/article/pii/S2405896317319432.

[9] Griffin Francis et al. 'Sampling-Based Trajectory Generation for Autonomous Spacecraft Rendezvous and Docking'. In: *AIAA Guidance, Navigation, and Control (GNC) Conference*. DOI: 10.2514/6.2013-4549. eprint: https://arc.aiaa.org/doi/pdf/10.2514/6.2013-4549. URL: https://arc.aiaa.org/doi/abs/10.2514/6.2013-4549.

[10] D.A. Surovik and D. Scheeres. 'Adaptive envisioning of reachable mission outcomes for autonomous motion planning at small bodies'. In: *Advances in the Astronautical Sciences* 150 (Jan. 2014), pp. 537–551.

[11] Francesco Capolupo and Pierre Labourdette. 'Receding-Horizon Trajectory Planning Algorithm for Passively Safe On-Orbit Inspection Missions'. In: *Journal of Guidance, Control, and Dynamics* 42.5 (2019), pp. 1023–1032. DOI: 10.2514/1.G003736. eprint: https://doi.org/10.2514/1.G003736. URL: https://doi.org/10.2514/1.G003736.

[12] L. Summerer R. Schonenborg O. Dubois-Matra E. Luraschi A. Cropp H. Krag K. Wormnes R. L. Letty and J. Delaval. 'ESA technologies for space debris remediation'. In: Proceedings of the 6th IAASS Conference: Safety is Not an Option, 2013, pp. 3–4.

[13] David A. Surovik and Daniel J. Scheeres. 'Adaptive Reachability Analysis to Achieve Mission Objectives in Strongly Non-Keplerian Systems'. In: *Journal of Guidance, Control, and Dynamics* 38.3 (2015), pp. 468–477. DOI: 10.2514/1.G000620. eprint: https://doi.org/10.2514/1.G000620. URL: https://doi.org/10.2514/1.G000620.

[14] David A. Surovik and Daniel J. Scheeres. 'Autonomous Maneuver Planning at Small Bodies via Mission Objective Reachability Analysis'. In: *AIAA/AAS Astrodynamics Specialist Conference*. DOI: 10.2514/6.2014-4147. eprint: https://arc.aiaa.org/doi/pdf/10.2514/6.2014-4147. URL: https://arc.aiaa.org/doi/abs/10.2514/6.2014-4147.

[15] E. Komendera, D. Scheeres and E. Bradley. 'Intelligent Computation of Reachability Sets for Space Missions'. In: *IAAI*. 2012.

[16] Damion Dunlap, Emmanuel Collins and Charmane Caldwell. 'Sampling Based Model Predictive Control with Application to Autonomous Vehicle Guidance'. In: (Jan. 2008).

[17] E. Komendera. 'Description of the Reachability Set Adaptive Mesh Algorithm ; CU-CS-1090-12'. In: 2012.

[18] Christian Grimme. *Picking a Uniformly Random Point from an Arbitrary Simplex*. Jan. 2015. DOI: 10.13140/RG.2.1.3807.6968.