Project Number 101017258

D2.4 Multi-Robot Monitoring Online Trajectory Generation

Version 1.0
23 December 2022
Final

Public Distribution

University of Luxembourg

Project Partners: Aero41, ATB, AVL, Bonn-Rhein-Sieg University, Cyprus Civil Defence, Domaine Kox, FORTH, Fraunhofer IESE, KIOS, KUKA Assembly & Test, Locomotec, Luxsense, The Open Group, Technology Transfer Systems, University of Hull, University of Luxembourg, University of York

Project Partner Contact Information

Aero41 Frédéric Hemmeler Chemin de Mornex 3 1003 Lausanne Switzerland E-mail: frederic.hemmeler@aero41.ch	**ATB** Sebastian Scholze Wiener Strasse 1 28359 Bremen Germany E-mail: scholze@atb-bremen.de
AVL Martin Weinzerl Hans-List-Platz 1 8020 Graz Austria E-mail: martin.weinzerl@avl.com	**Bonn-Rhein-Sieg University** Nico Hochgeschwender Grantham-Allee 20 53757 Sankt Augustin Germany E-mail: nico.hochgeschwender@h-brs.de
Cyprus Civil Defence Eftychia Stokkou Cyprus Ministry of Interior 1453 Lefkosia Cyprus E-mail: estokkou@cd.moi.gov.cy	**Domaine Kox** Corinne Kox 6 Rue des Prés 5561 Remich Luxembourg E-mail: corinne@domainekox.lu
FORTH Sotiris Ioannidis N Plastira Str 100 70013 Heraklion Greece E-mail: sotiris@ics.forth.gr	**Fraunhofer IESE** Daniel Schneider Fraunhofer-Platz 1 67663 Kaiserslautern Germany E-mail: daniel.schneider@iese.fraunhofer.de
KIOS Maria Michael 1 Panepistimiou Avenue 2109 Aglatzia, Nicosia Cyprus E-mail: mmichael@ucy.ac.cy	**KUKA Assembly & Test** Michael Laackmann Uhthoffstrasse 1 28757 Bremen Germany E-mail: michael.laackmann@kuka.com
Locomotec Sebastian Blumenthal Bergiusstrasse 15 86199 Augsburg Germany E-mail: blumenthal@locomotec.com	**Luxsense** Gilles Rock 85-87 Parc d'Activités 8303 Luxembourg Luxembourg E-mail: gilles.rock@luxsense.lu
The Open Group Scott Hansen Rond Point Schuman 6, 5th Floor 1040 Brussels Belgium E-mail: s.hansen@opengroup.org	**Technology Transfer Systems** Paolo Pedrazzoli Via Francesco d'Ovidio, 3 20131 Milano Italy E-mail: pedrazzoli@ttsnetwork.com
University of Hull Yiannis Papadopoulos Cottingham Road Hull HU6 7TQ United Kingdom E-mail: y.i.papadopoulos@hull.ac.uk	**University of Luxembourg** Miguel Olivares Mendez 2 Avenue de l'Universite 4365 Esch-sur-Alzette Luxembourg E-mail: miguel.olivaresmendez@uni.lu
University of York Simos Gerasimou & Nicholas Matragkas Deramore Lane York YO10 5GH United Kingdom E-mail: simos.gerasimou@york.ac.uk nicholas.matragkas@york.ac.uk	

Document Control

Version	Status	Date
0.1	Document outline	19 July 2022
0.2	Initial draft	30 July 2022
0.3	First draft	8 August 2022
0.4	Internal reviews	8 December 2022
0.5	Internal reviews updates	15 December 2022
0.6	Final version for partner reviews	19 December 2022
0.7	Final version partner reviews updates	21 December 2022
0.8	Further final version partner reviews updates	21 December 2022
1.0	Final QA version	23 December 2022

Table of Contents

List of Figures

List of Tables

Acronyms

EU	European Union
ExSce	Executable Scenario
MPC	Model Predictive Control
MRS	Multi-Robot Systems
SESAME	Secure and Safe Multi-Robot Systems
SnT	Interdisciplinary Centre for Security Reliability and Trust
WP	Work Package
EDDI	Executable Digital Dependability Identities
ASTC	Adaptive Super-Twisting Controller

Executive Summary

This deliverable reports the Perception-Aware Trajectory Planning and Tracking for Multi-Robot Systems (MRS) in Task 2.4 of Secure and Safe Multi-Robot Systems (SESAME) project. In this report, we develop an online trajectory generation approach that minimises the uncertainty of a group of robots while maximising their efficiency in performing specific MRS tasks. Efficiency can be seen as safe navigation to the target point or fulfilling the task. Also, this task will develop a control approach to track the generated trajectory. We consider the close environment information gathered by nearby robots using the collaborative sensor fusion approach, provided in Task 2.3 and the mission goals.

Tackling trajectory planning, an online trajectory optimization approach is developed to compute the fastest trajectory. We use Ipopt for solving the optimization problem. We implement some improvements in the modelling of the optimization problem to speed up the solver. The obstacle avoidance issue is considered as a safety region. Furthermore, the position and field of view are retained in the given area, satisfying the maximization of capturing the perceptive information. The tracking control design is tackled in the presence of external disturbance using adaptive super-twisting control. The proposed controller provides the finite time convergence and attenuation of the disturbance. The algorithms and instructions to execute the proposed approaches are given. The efficiency of the proposed approaches is investigated by high-fidelity simulations in Gazebo as well as experimental studies.

1 Introduction

In this section, the preliminary remarks on the project and the main research objectives of this deliverable are elaborated. Accordingly, the main research directions are identified as a set of detailed questions, which are mathematically formulated, and corresponding solutions are presented in the proceeding sections. Moreover, the relationship between the overall project aims and the proposed solution is established. Accordingly, the contributions and novelties are summarized.

Trajectory planning is one of the most important capabilities of European Union (EU) SESAME project. For instance, aerial robots for autonomous pest management in viticulture, or ground robots for disinfecting hospital environments, trajectory planning determines the trajectory to be followed by the robots. This is to find an optimal path to the destination for MRS [1]. Consequently, optimal decisions need to be taken for various mission-critical operations [2]. Accordingly, the trajectory can be defined as "*a time parameterized motion reference, i.e., geometric values of position, heading, derivatives associated with time law, passing through the waypoints, considering geometrical feasibility, collision avoidance, kinematics and dynamics* " [1]. One of the main objectives is to integrate perception awareness into MRS trajectory planning, i.e., the construction of the trajectory on which the perception metric is incorporated. The perception can be defined as "gathered information on the MRS dynamics, or the environment, i.e., localization, geometric mapping, semantic mapping, obstacle detection, texture, etc" [3, 4]. Therefore, given the use case and corresponding demands, we require to present an accurate definition of perception awareness. After a thorough literature review, this can be defined as "planning the motion commands to achieve a goal (e.g., reach a point) taking into consideration some restrictions/limitations/features of the perception". A "mathematical relation between perception, sensing, and corresponding actions that MRS take" is required. Indeed, it can be stated as collaborative and perception-aware trajectory planning, which minimizes environmental uncertainty by using data provided by other robots in a team combined with data collected by the perception components of the robot. Also, novel collaborative perception and sensor fusion algorithms allow robots to have an accurate representation of their environment by using the data of their peers. It is worth noting that perception awareness is not to be confused with active perception, i.e., planning the motion commands to maximize the performance of the perception, which is not our task. It should be noted that even though a generic scheme is of interest, "perception awareness is to be addressed for some particular use-cases, with a demonstration of the applicability for others".

One of the main features of the trajectory planner is collision avoidance with other obstacles, humans, and other robots. These, generally, can be titled as static and dynamic obstacles [4]. So, the information on the obstacles is to be implemented into the proposed solution. Furthermore, safety and security are to be considered in the planning. In some works, collision avoidance is considered as the geometrical feasibility of the planning. So, we need to implement the updated information on the obstacles into the proposed planner. One trivial question is how to measure/estimate the information of obstacles. Also, is there priori knowledge about the obstacles to be considered? Also, a formation performance might be requested from the MRS. Moreover, perception awareness (localization uncertainty, mapping uncertainty, semantic information) and safety/security (in a very broad sense) are to be considered in the planning.

Another distinct feature to be implemented is the concept of situational awareness. This may include localization, geometric mapping, semantic mapping, obstacle detection and tracking, etc [5–7]. So, we take into account the mathematical relation between perception, sensing, and corresponding actions that MRS take. Indeed, it can be stated as collaborative and perception-aware trajectory planning, which minimizes environmental uncertainty by using data provided by other robots in a team combined with data collected by the perception components of the robot.

In MRS there might be different robot types involved, e.g., manipulators, ground mobile robots or UAVs. The generic model enables us to describe high-level robotic capabilities (e.g., fly, grasp, see), skills (e.g., what the robot is good at), roles (e.g., tracker, mapper) and behaviours (e.g., mapping, planning, orbiting) based on the type of the robot and its onboard sensing devices. Since we are dealing with different use cases with different demands, the robot type is to be foreseen in the scheme. However, it might narrow the application of the proposed solution. So, we should make a tradeoff between the flexibility of the proposed solution (generalizability of the scheme on versatile generic models) and the applicability of a given use case. One way to resolve this is to present a potentially versatile solution for one use case with potential application/extension on the others with some modifications, to some extent. This is to be pointed out to the other partners. Indeed, use-case partners need to adapt the proposed versatile scheme to their use cases.

Most planning approaches are constructed as an on/offline optimization problem, in which the trajectory with the optimized feature is obtained. More importantly, the aspect we address is the structural design of planning and tracking parts, if we address and design them separately or simultaneously, considering the priority. This implies another preliminary step to be taken, as "*the general planning and tracking structural scheme, for the given MRS and use-cases, addressing the parts to be (de)centralized and on/offline*". Mostly, the time that the robot takes to track the trajectory between two given points is minimized. So, we need to come up with an answer to "what is the optimization index in the planning; time, energy, snap, perception ...?" Also, for optimization, an initial trajectory is required, e.g., computed analytically. In terms of technical development of the project, as mentioned in recent publications, to construct the above-mentioned optimization problem, the RMS is to be mathematically modelled with the coupled dynamics. More importantly, considering the computational burden, the model is to be as simple as possible, e.g., a linear model. However, regardless of the inexpensive computational burden of simple models, this imposes model uncertainty and mismatch in practice. Accordingly, our preliminary need is "kinematics of the MRS given the robot types as well as the simplified dynamics of the robot, i.e., first-order model approximation, with identification of the potential model uncertainty and exogenous disturbance". Regarding the project objective, we use a generic simple model that can be potentially extended to a more complex one, to focus on the higher-level task and corresponding requirements.

In addition to the trajectory planning, one significant capability is the trajectory tracking control, i.e., the design of the control commands to make the MRS stable as well as to track the trajectory as close as possible. This can be defined as "the desired commands to make the dynamics follow the trajectory" [4]. The tracking task satisfies the desired position, velocity and orientation while fulfilling the given task. This can be defined as controlling the MRS to track the planned trajectory as close as possible. Also, the issue of stability is to be addressed. The latter is significant considering the model uncertainty [8,9]. In terms of feasibility and reliability, taking into account the prior research activities and experience available in the Interdisciplinary Centre for Security Reliability and Trust (SnT), Model Predictive Control (MPC) is the feasible and reliable solution. On the other hand, for sake of the novel research activities, the use of state-of-the-art modern nonlinear controls is sought. Consequently, our next research objective is to "Tracking control type, i.e., incorporation of MPC in tracking part and considering other advanced controls in parallel, tackling stability and model uncertainty".

Finally, within the framework of autonomous robotic systems, the features of safety and security are vital. However, considering the available literature, the devised approaches mainly stem from the corresponding definitions of safety and security, i.e., the definition implies the corresponding solution. So, "the definition of safety and security factors, conceptually and mathematically, within planning or tracking" implies the corresponding solution. Security can be the resilience of the solution to unauthorized access to communication channels. Security might be foreseen as the resilience of the solution to unauthorized access to communication channels to be foreseen in planning tasks. On the other hand, in terms of safety, can be interpreted as tolerance against a family of faults, avoidance from a given area, restriction of MRS states, and safely bound on the tracking drift, which can be foreseen in the tracking task. Cumulatively, we can present the solution as reliable autonomous robotic systems with quality assurance, risk assessment and trust level. Therefore, "the definition, mathematically and conceptually, of quality assurance, risk assessment and trust criteria" are needed. [10,11].

The main goal of this deliverable is to present the proposed solution, addressing the above-mentioned aspects of the versatility of MRS tasks.

1.1 Document Purpose

This document is prepared in the context of the SESAME project. More precisely, it refers to Task 2.4: "Perception-Aware Trajectory Planning and Tracking for MRS" of Work Package (WP) 2: Sensor Fusion and Collaborative Intelligence. This task will develop an online trajectory generation approach that minimises the uncertainty of a group of robots while maximising their efficiency in performing specific MRS tasks. Also, this task will develop a control approach to track the generated trajectory based on the model predictive control approach, developed in Task 2.2. We will focus on the provision of sensor feedback information among the members of a collaborative robotic team at the same time that robots perform individually or collaboratively specific tasks. Online perception-aware trajectory generation will consider the close environment information gathered by nearby robots using the collaborative sensor fusion approach developed in Task 2.3 and the mission goals. Every robot will be "monitored" by one or more robots, but not all of them. Also, when a cyber-attack or sensor malfunction affects one or more members of the team, non-attacked robots will generate a rescue trajectory providing further sensor feedback to the compromised robots and helping them to maintain their operational and safe state.

This is the fourth deliverable of the WP2, "D2.4: Multi-Robot Monitoring Online Trajectory Generation". This deliverable will report on work performed in Task 2.4, including the novel approach to online collaborative trajectory planning which reduces the uncertainty of situational awareness. Also, the deliverable will provide the analysis and validation of the experiments developed in simulated environments and in-the-lab settings. It provides in-depth details of the algorithms behind the Multi-Robot Monitoring Online Trajectory Generation. And finally, it provides some early results regarding the performance of the different algorithms. The component development process follows an engineering systems design approach.

In this report, the requirements for this task are initially introduced and a systematic scheme is provided to identify the parts to be accomplished and corresponding inputs from the other partners. Also, the evaluation plan to assess the requirements, as well as the project objectives, are presented, accordingly. It should be noted that the sequence by which the requirements are introduced does not necessarily imply the procedure, i.e., some requirements can be done earlier or simultaneously, considering the priority. Moreover, we address which part is on/offline and (de)centralized. More importantly, how to integrate online new information. Therefore, we need to tackle this as a "Sequential diagram of the overall procedure, high/low-level parts of the solution to fulfil the following requirements". In summary, the overall objectives are as follows.

- Online trajectory generation approach that minimizes the uncertainty of MRS, to fulfil the overall common mission goals in offline/online, (de)centralized ways.
- incorporating the environment information using Task 2.3.
- MPC and novel nonlinear control approach for tracking the generated trajectory.
- Cyber-attack or sensor malfunction affecting non-attacked robots will generate a rescue trajectory providing further sensor feedback to maintain an operational and safe state.

Moreover, the proposed solutions are evaluated as

- Numerical and in-lab experimental evaluation of the planner, satisfying the metrics, the computational time/burden, and situational/perception awareness.
- Numerical and in-lab experimental evaluation of the tracking control, satisfying the metrics, investigating the closed-loop system, in terms of stability and tracking error.
- Safety, security and quality assurance for a tentative trajectory.
- We provide the components in the form of algorithms with demonstrations of drones as an example.

Finally, it should be noted that the practical implementation and integration on use cases are sought in WP8. In this deliverable, we provide high-fidelity simulations and experiments to validate the proposed approaches as the proof of concept. Furthermore, the potential applicability of the proposed approaches on the use cases is briefly motivated in Section 6.

1.2 Relationship to other Deliverables

Considering the above-mentioned points, Table 1 presents the inputs which are provided to us by other partners. It should be noted that to avoid any interruption, we could take these inputs for granted, if possible, to continue our research. Also, Table 1 elaborates the outputs, expected from us. This is illustrated in Figure 1. In summary, the inputs to our component are:

- Robots model, parameters, dynamic and kinematic restrictions (by Executable Scenario (ExSce)),
- Information of the sensors of the robots (by ExSce),
- The estimated state of robots (i.e., pose, velocity, etc.), non-robotic agents (e.g., humans), and metric-semantic model of the environment with stochastic information (e.g., the covariance of the estimation), (by Collaborative perception).
- Task plans information, including start and endpoints, the intermediate regions of interest, dependencies between the different robots, and temporal dependencies between the subtasks or dependencies between the states of the robot,
- Safety, security and quality assurance metrics, to be mathematically well-defined, (by ExSce, Collaborative intelligence and Executable Digital Dependability Identities (EDDI)).

Moreover, the expected outputs are

- Planned trajectories for each individual robot, including time-parameterized motion references for each robot, safety, security and quality assurance metrics were achieved for each planned trajectory,
- Robot commands in the form of actuator/driver command to the robotics platforms e.g. desired velocity commands,
- Algorithms of Higher level centralized planner, Lower level decentralized planner and Lower level decentralized control,
- Non-attacked robots will generate a rescue trajectory providing further sensor feedback to the compromised ones,
- Numerical simulations and experimental studies on in-lab drones as an example.

Accordingly, the interfaces to exchange information with other components are identified as

- The dynamics and kinematics of the robots with the ExSce,
- The sensors of the robots with the ExSce,
- Task plans with the ExSce and/or Collaborative Intelligence, including, requirements, tasks, temporal constraints, start and endpoints,
- The MRS system structure and the possibility to exchange information among the robots or with a control station
- Complete situational awareness with Collaborative Perception,
- Safety, security and quality assurance metrics for a tentative trajectory with the EDDI,
- Robot-agnostic interface for the planned trajectories,
- Robot-agnostic interface for the actuator/driver commands to the robotics platforms.

1.3 Contribution and Novelties

Our main focus is on MRS, where the trajectories of several robots need to be planned in a coordinated way, in general, to fulfil the overall common mission goals. The robots have to fulfil local tasks, such as the movement from a given start- to an endpoint, while for instance visiting different regions of interest in between and performing actions herein under temporal constraints. Furthermore, there might be dependencies between the robots and their dynamic states such as keeping a formation or avoiding collisions among each other. Therefore, task and trajectory planning are closely related and sometimes even done simultaneously. There are many approaches to performing MRS trajectory planning, mainly categorized as offline vs online, and centralized

Table 1: Inputs and outputs of the component

Input/output	From/to	Data
Input (LU6)	ExSce, BRUS-9	Dynamics/kinematics of the robots: Information of the robot dynamics and kinematic, dynamic and kinematics limitations
Input (LU7)	Use Case	Task plans (before launching): Robot and team capabilities, high-level robotic capabilities (e.g., fly, grasp, see), skills (e.g., what the robot is good at), roles (e.g., tracker, mapper) and behaviours (e.g., mapping, planning), Trajectory start/endpoints, the intermediate regions of interest, and dependencies between robots, required formations or collision avoidance, or state constraints of the robots.
Input (LU8)	Multi-Agent System	Task plans (at run time): New waypoints
Input (LU9)	ExSce	Task decomposition and allocation
Input (LU10)	Collaborative Perception	Sensing and perception, gathered environmental model: estimated states of each robot, non-robotic, and metric-semantic model of the environment with stochastic information
Input (LU11)	Runtime EDDI	Metric requests: Desired metrics on safety, security, and quality assurance for each task plan
Output (LU12)	MRS (simulation)	Planned trajectories: Feasible (collision-free) considering the kinematic/dynamic of robotic, with real-time re-planning with perception-awareness and risk-awareness.
Output (LU13)	MRS (simulation)	Actuator/driver commands to the robotics platforms.
Output (LU14)	Runtime EDDI	Metrics achieved: The planned trajectories achieve the given metrics on safety, security, and quality assurance metrics

vs distributed/decentralized planning. The optimal approach to planning the trajectory depends mainly on the overall mission requirements, the MRS system structure and the possibility to exchange information among the robots or with a control station.

The trajectory planning is also ruled by the environment of the MRS and the model of the environment available for the planning, either in a centralized form or as partial models in single robots. The environment consists of static structures, static or dynamic objects and the free space in between. The modelling of the environment/situational awareness can be based on many different levels of abstraction and include different levels of uncertainty. The environmental model is continuously updated using mainly the sensing and perception of the different robots, equipped with different types of onboard sensors. Therefore, trajectory planning and tracking are closely linked to environmental modelling as well as the MRS sensing and perception capabilities.

However, the robotic motion does not only depend on the sensing and perception but the perceptual processing itself is influenced by intended actions (active perception). Recent research has extended active perception with support for partial or full mission “planification” by generating perception-aware path and trajectory plans. Advances in the area focus on minimising localisation uncertainty by simultaneously updating the path planning using the richness of texture information in the environment, and on minimising state estimation uncertainty by computing the feasibility of trajectory-planning and trajectory-tracking based on the kinematic and dynamic models of the robot. In MRS, performing perception-aware trajectory planning in a collaborative but distributed way is a further challenge, strongly related to the coordination and communication schemes during the planning.

Finally, additional challenges for the MRS trajectory planning and tracking arise from the requirements of safety and security. On the other hand, unauthorized access to the communication system of the MRS could be used to compromise the information that is exchanged between the robots during the trajectory planning, leading to a decrease in performance or even a failure or damage to the MRS or the environment. In our presented component, we provide a capability for MRS trajectory planning and tracking that takes the aforementioned challenges into account. We will develop a cascaded solution, providing online trajectory planning with continuous re-planning, and online trajectory tracking.

In our proposed approach, the trajectory planning part will include a high-level centralized planner, where the global MRS trajectory planning problem is formulated as one overall optimization problem computing the rough trajectories that each robot agent is to track. In a cascade and in a distributed way, each robotic agent will rely on a planner to compute its detailed trajectory to be tracked, based on the given rough one. The trajectory tracking will guarantee that the previously planned detailed trajectories are tracked with no deviations, providing the actuator/driver commands to the robotics platforms, e.g., desired velocity commands. We will apply model predictive control (MPC) in a decentralized way, e.g., each robot independently tracks its own trajectory. The following aspects will be considered at different levels on each component, i.e. planner/tracker: (1) collision avoidance with the structures, static or dynamic objects and the other robots, also including dynamic and stochastic models of the potential obstacles (2) kinematic and dynamic limitations of the robots, as well as their model uncertainties and the possibility for online parameter estimation/update, (3) capabilities and constraints of the different robotic sensors and the limits of the situational awareness algorithms to provide perception-aware planning, (4) safety, security, and quality assurance aspects, e.g. by including risk and trust in the planning, generated by the EDDI components at runtime. Herein, safety-related risk could be related to the environmental situation, the uncertainty of observations or the consequences of robotic actions.

The main contributions are as follows.

- The trajectory planning problem will be conveniently split into an MRS high-level centralized part and a low-level distributed part, achieving a real-time operation and robust performance.
- The distributed trajectory planning and tracking components will be heterogeneous, i.e., different for each robotics platform, while the centralized trajectory planner will be generic and versatile to include all the targeted robotics platforms, e.g., different kinematic and dynamic models and restrictions. Also, the overall problem formulation is given on a generic dynamic, considering the different use cases with

different robot types and the component is designed in a modular way to let each part usable with the least modification required.
- The information provided by the perception, sensing, and situational awareness components will be exploited at the previously mentioned three different levels, i.e., centralized planner, distributed planner, and distributed trackers.
- Safety and security aspects will be considered at the previously mentioned three different levels, i.e., centralized planner, distributed planner, and distributed trackers.

1.4 Document Structure

The rest of this deliverable is organized as follows. In Section 2, we review the state-of-the-art approaches to highlight the contributions of the proposed solution. In Section 3, the trajectory planning and tracking problem is formulated for generic dynamics and the corresponding approach to tackle this problem is presented. Both numerical simulation and experimental results are studied in Section 4. Then, the extension of the proposed solution is considered in Section 5. The potential applicability of the proposed approaches for use cases is described in Section 6. The concluding remarks are given in Section 7. In Appendix A, implementation remarks of the designed codes for drone deployment are given.

2 Related Work

This section discusses the related works and critically reviews similar approaches for trajectory planning and tracking. This is to highlight the contributions of the proposed solution.

Recent research on perception-aware trajectory planning has extended active perception with support for partial or full mission 'planification' by generating perception-aware path and trajectory plans. Advances in the area focus on minimizing localization uncertainty by simultaneously updating the path planning using the richness of texture information in the environment [12], and on minimizing state estimation uncertainty by computing the feasibility of trajectory-planning and trajectory-tracking based on the kinematic and dynamic models of the robot [3]. The generation of risk-aware trajectories using the uncertainty corresponding to the quality of observations is also explored [13]. Trajectory planning has been widely investigated for drones. However, with some minor modifications, the existing approaches can be extended for the different robotic systems.

2.1 Hard-constrained methods

Hard-constrained methods are pioneered by minimum-snap trajectory [14], in which piecewise polynomial trajectories are generated through quadratic programming (QP). Richter et al. [15] presented a closed-form solution to minimum snap trajectories. [16] generates trajectories in a two-step pipeline. Free space represented by a sequence of cubes [16], spheres [17] or polyhedrons [18] is firstly extracted, which is followed by convex optimization, which generates a smooth trajectory within the feasible space. Safe regions around initial paths are extracted as convex flight corridors, within which QP is solved to generate smooth and safe trajectories. Among these methods, poorly chosen time allocation of piecewise polynomials usually leads to unsatisfying results. To this end, fast marching [19] and kinodynamic search [20] are utilized to search for initial paths with more reasonable time allocation. Gao et al. [19] also proposed to represent trajectories as piecewise Bézier curves so that safety and dynamical feasibility are guaranteed. They proposed a method to search for a path with well-allocated time and guarantee the safety and kinodynamic feasibility of trajectory through optimization. Hard-constrained methods ensure global optimality by the convex formulation. However, distance to obstacles in the free space is ignored, which often results in trajectories being close to obstacles. Besides, the kinodynamic constraints are conservative, making the trajectory's speed deficient for fast flight. Tordesillas et al. [21] adopted a mixed-integer QP (MIQP) formulation to find a more reasonable time allocation of the trajectory. [22] proposed a B-spline-based kinodynamic search to find an initial trajectory which is then refined by an elastic band optimization approach. The use of a uniform B-spline ensures dynamic feasibility but could generate conservative motion. One common drawback of these methods is that the time allocation of the trajectory is given by naive heuristics. However, a poorly chosen time allocation significantly reduces the quality of the trajectory. Besides, a feasible solution can only be obtained by iteratively adding more constraints and solving the quadratic programming problem, which is undesirable for real-time applications.

2.2 Soft-constrained methods

Soft-constrained methods are also methods formulating trajectory generation as a non-linear optimization problem that takes smoothness and safety into account. [23] generates discrete-time trajectories by minimizing its smoothness and collision costs using gradient descent methods. [24] has a similar problem formulation, but the optimization is solved by a gradient-free sampling method. [25] extended them to continuous-time polynomial trajectories. Since the time parameterization is continuous, it avoids numeric differentiation errors and is more accurate to represent the motions of quadrotors. However, it suffers from a low success rate. To solve this problem, [26] finds a collision-free initial path firstly using an informed sampling-based path searching method. This path serves as a higher-quality initial guess of non-linear optimization and thus improves the success rate. In [27], the trajectory is parameterized as a uniform B-spline. Since a B-spline is continuous by nature, there is no need to enforce continuity explicitly, which reduces the number of constraints. It is also particularly useful for local replanning thanks to its property of locality. Soft-constrained methods utilize gradient information to push trajectories far from obstacles but suffer from local minima and have no strong guarantee of success rate and kinodynamic feasibility. Our optimization method also utilizes gradient information to improve the safety of the trajectory. However, unlike previous methods in which computational expensive line integrals along the trajectory are calculated, the formulation is redesigned to be simpler based on the convex hull property of B-spline. It greatly improves the computation efficiency as well as the convergent rate.

2.3 Gradient-based trajectory optimization

Gradient-based trajectory optimization [28] is one of the major trajectory generation approaches, which formulates the problem as a non-linear optimization that minimizes an objective function. Gradient-based motion planning is the mainstream for quadrotor local planning. Based on the pioneering works [24,29] that formulate the local planning problem as unconstrained nonlinear optimizations, a series of works (see [30] and reference therein) are proposed. They consider the smoothness, feasibility, and safety of the trajectory using various parameterization methods, including polynomial and B-spline. Recently, it is proposed a single-quadrotor navigation system named EGOPlanner [31], further reduces computation time using a more compact environment representation. Recent works [28] revealed that they are particularly effective for local replanning, which is a key component for high-speed flight in unknown environments. GTO methods are preferable for replanning due to their high efficiency. However, their local minima issue may lead to undesired solutions. Interest in this method was revived recently by [23], which generates discrete-time trajectories by minimizing its smoothness and collision costs using covariant gradient descent. [24] has a similar formulation but solves the problem by sampling neighbouring candidates iteratively. The stochastic sampling strategy partially overcomes the local minima issue but is computationally intensive. [25] extended the method to continuous-time polynomial trajectories to avoid differential errors. It also does random trajectory perturbation and optimization restart for a higher success rate to slightly relieve the typical local minima issue of such methods. However, the improvement is insignificant. [26] improved the success rate by providing a high-quality initial path, which is found by an informed sampling-based path searching. However, due to insufficient success rate and efficiency, it only applies to low-speed flights. In [27], the trajectory is parameterized as a uniform B-spline. It showed that the continuity and locality properties of the B-spline are particularly useful for trajectory replanning. to fly at a moderate speed. [32] further exploited the convex hull property of B-spline and improve the optimization efficiency and robustness by a large margin. However, given a poor initial trajectory in complex environments, this method still suffers. As a result, [33] adopts an iterative post-process to improve the practical success rate of [32]. By far, local minima still remain a challenge, since no method copes with it essentially. In this paper [28], we propose path-guided optimization PGO, which incorporates a geometric path in the optimization. As the path effectually guides the optimization to escape from infeasible local minima, the planning success rate is guaranteed. Moreover, multiple distinct paths produced by the topological path searching are integrated with the PGO to seek plentiful locally optimal solutions, which ensures higher trajectory quality.

2.4 Topological path planning

Topological path planning has been works utilizing the idea of topologically distinct paths for planning, in which paths belonging to different homotopy (homology) [34] or visibility deformation [35] classes are sought. Topological planning is used to escape local minima. Based on the homology equivalence relation in 2-D surfaces originated from complex analysis [36]. Rosmann et al. [37] present a trajectory planning method in distinctive topologies using Voronoi and sampling-based front-ends and TEB (Timed-Elastic-Bands) local planner [38] as back-ends. However, homology equivalence relation in 3-D is far more trivial. To capture distinctive useful paths, Jaillet et al. [35] construct visibility deformation roadmaps that encode richer and more relevant information than representative paths of the homotopy classes. Based on [35], Zhou et al. [28]

enable real-time topological planning by proposing an efficient topology equivalence checking. We extend EGO-Planner to accelerate the front end for topological planning further. [34] constructs a variant of the probabilistic roadmap (PRM) to capture homotopy classes, in which path searching and redundant path filtering are conducted simultaneously. In contrast, [37,39] firstly creates a PRM or Voronoi diagram, after which a homology equivalence relation based on complex analysis is adopted to filter out redundant paths. These methods only apply to 2-D scenarios. To seek 3-D homology classes, [36] exploited the theory of electromagnetism and propose a 3-D homology equivalence relation. Capturing only homotopy classes in 3-D space is insufficient to encode the set of useful paths, as indicated in [35], since 3-D homotopic paths may be too hard to deform into each other. To this end, [35] leverages a visibility deformation roadmap to search for a richer set of useful paths. [40,41] convert maps built from SLAM systems into sparse graphs representing the topological structure of the environments. [35,40,41] focus on global offline planning and are too time-consuming for online usage. Topological path searching in [28,42] is conceptually closest to [35], but with a reinvented algorithm for real-time performance. They extend the concept of visibility deformation to make it computationally cheaper and introduce a path-shortening and pruning technique to maintain compact sets of useful paths.

2.5 Collision avoidance

Several approaches exist for collision avoidance in dynamic environments, which include velocity obstacles [43], decentralized NMPC [44] and sequential NMPC [45]. However, these approaches were deterministic and did not account for uncertainties in perception and motion. The concept of velocity obstacles was extended to handle motion uncertainties by using conservative bounding volumes [46]. Yet, the robot dynamics were not fully modelled and the motion was limited by planning a constant velocity motion. These issues can be overcome by using NMPC for planning. [47] introduced a decentralized NMPC where robot motion uncertainties were taken into account by enlarging the robots with their 3-sigma confidence ellipsoids. However, bounding volumes can be conservative and lead to infeasible solutions in cluttered environments. In [48] we explicitly consider the collision probability and formulate a chance-constrained NMPC problem. A chance-constrained MPC problem was formulated by [49] for systems with linear dynamics and planar motion, where rectangular regions were computed and overlap avoided in a centralized mixed integer program formulation. The approach in [48] is not centralized and can be applied to robots with nonlinear dynamics navigating in three-dimensional spaces. If one assumes set-bounded motion uncertainty models, then robust MPC can be employed to plan safe trajectories [50], or a guaranteed trajectory tracking error bound [51] can be used. However, uncertainties described by Gaussian probability distributions, such as those resulting from Kalman filters, are unbounded. If we consider Gaussian distributions, then objects can be approximated by larger bounding volumes that correspond to sigma hulls [52] which are based on confidence levels. With this method, collision checking can be performed very fast. However, the enlarged bounding volumes generally overestimate the collision probability [53]. Hence, when navigating in cluttered environments, the approach tends to lead to sub-optimal or infeasible solutions [54]. By assuming a constant probability density of the robot's position within the obstacle, the collision probability can be approximated by the density multiplied by the volume occupied by the obstacle. [55] uses the probability density of the centre of the obstacle, while [53] uses the maximum density on the surface of the obstacle to provide an upper bound of collision probability. Both methods are fast, but they only work well when the sizes of objects are relatively very small compared with their position uncertainties. The collision probability can be computed directly via sampling [56]. However, this is computationally intensive and thus not eligible for real-time collision avoidance. Another alternative is to consider convex polygonal obstacles [57]. Under the assumption that object positions follow Gaussian distributions, the resulting linear chance constraints can be transformed directly into deterministic constraints of the mean and covariance of the positions. However polygonal obstacles are ill-posed for online constrained optimization, where smooth shapes are preferred to avoid local minima. In this letter, we consider spherical robots and ellipsoidal dynamic obstacles. We locally linearize the nonlinear collision avoidance constraints and the corresponding chance constraints are reformulated into deterministic constraints on the robot's state mean and covariance. Such a linearization technique was used for deterministic multi-agent collision avoidance [58]. We mathematically formalize its use in the context of probability-based stochastic collision avoidance.

2.6 Onboard sensory system for planning in unknown environment

To deal with unknown environments in navigation, different strategies have been used. Many methods adopt the optimistic assumption [17], which treats the unknown space as collision-free. This strategy improves the chance of reaching goals but may not guarantee safety. On the contrary, some other methods regard unknown space as unsafe and only allow motions within the known-free space [59] or sensor FOV [60]. In [60], the sensor FOV constraint is partially relaxed by choosing safe motion primitives generated in the past. Although these restrictions ensure safety, they lead to conservative motion. Recently, Tordesillas et al [61] proposed a strategy that plans in both the known-free and unknown space. Instead of being over-optimistic about the unknown space, it always maintains backup trajectories to ensure safety. The limitation of the above-mentioned strategies is the lack of environment perception awareness, which is of significant necessity in fast flight. Although much attention has been paid to planning with the awareness of localization [62] and target tracking [63], less emphasis is put on environment perception. Richter and Roy [64] proposed a learned heuristic function to guide the path searching into areas with greater visibility to unknown space, but it may not generalize well to complex 3D environments. Heiden et al. [65] showed an integrated mapping and planning framework for active perception. The planner iteratively simulates future measurements after executing specific motions, predicts the uncertainty of the map, and minimizes the replanning risk. Its main drawback is the prohibited runtime for online usage. In [66], a local planner is coupled with local exploration to safely navigate a cluttered environment. However, it conservatively selects intermediate goals within known-free space, which restricts the flight speed. In this article, we present a perception-aware strategy to ensure that unknown dangers can be discovered and avoided early. It guarantees safety and does not lead to conservative behaviours.

The shift from offboard to onboard sensing based on cameras resulted in an increased number of works trying to connect perception and action [63]. In [67], the authors proposed a method to compute minimum-time trajectories that take into account the limited field of view of a camera to guarantee the visibility of points of interest. Such a method requires the trajectory to be parametrized as a B-spline polynomial, constraining the kind of motion the robot can perform. Also, perception is included in the planning problem as a hard constraint, posing an upper bound to the agility of the robot since such constraints must be satisfied at all times. Furthermore, the velocity of the projection of the points of interest in the image is not taken into account. Finally, the algorithm was not suited for real-time control of a quadrotor and was only tested in simulations. In [68], the authors focused on combining visual serving with active Structure from Motion and proposed a solution to modify the trajectory of a camera in order to increase the quality of the reconstruction. In such a work, a trajectory for the tracked features in the image plane was required, and the null space of the visual servoing task was exploited in order to render it possible for the such a feature to track the desired trajectory. Furthermore, the authors did not consider the underactuation of the robot, which can significantly lower the performance of the overall task due to potentially conflicting dynamics and perception objectives. In [69, 70], information gain was used to bridge the gap between perception and action. In the first work, the authors tackled the problem of selecting trajectories that minimize the pose of uncertainty by driving the robot toward regions rich of texture. In the second work, a technique to minimize the uncertainty of a dense 3D reconstruction based on the scene appearance was proposed. In both works, however, near-hover quadrotor flight was considered, and the underactation of the platform was not taken into account. In [71] a hybrid visual servoing technique for differentially flat systems was presented. A polynomial parameterization of the flat outputs of the system was required, and due to the computational load required by the designed optimization framework, an optimal trajectory was computed in advance and never replanned. This did not allow for coping with external disturbances and unmodelled dynamics, which during the execution of the trajectory can lead to behaviours different from the expected one. In [45], a real-time motion planning method for aerial videography was presented. In these works, the main goal was to optimize the viewpoint of a pan-tilt camera carried by an aerial robot in order to improve the quality of the video recordings. Both works were mainly targeted to cinematography, therefore they considered objectives such as the size of a target of interest and its visibility. Conversely, we target robotic sensing and consider objectives aimed at facilitating vision-based perception. In [72], the authors proposed a two-step approach for target-aware visual navigation. First, position-based visual servoing was exploited to find a trajectory minimizing the reprojection error of a landmark of interest. Then, a model predictive control pipeline was used to track such a trajectory. Conversely, [63] solves the trajectory optimization and tracking within a single framework. Additionally, that work only aimed at rendering the target visible but did not take into account that, due to the motion of the camera, it might not be detectable because of motion blur. We cope with this problem by considering in the optimization problem the velocity of the projection of the point of interest in the image plane.

3 Perception-Aware Trajectory Planning and Tracking: Algorithm

In this section, trajectory planner and tracker problems are formulated. Then, the corresponding solutions are presented. We tackle this as generic as possible. Accordingly, the use cases should opt for and specify components for the respective practice.

3.1 Overall architecture

Trajectory planning is one of the most important problems to be addressed to find an optimal path to the destination. Although in literature, a lot of research proposals exist on the path planning problems, still issues of target location, safe (collision-free) shortest possible path and identification persist keeping in view of the high mobility of MRS. Accordingly, optimal decisions need to be taken for various mission-critical operations. These decisions require a map or graph of the mission environment so that MRS is aware of its locations with respect to the map or graph. However, in the case the map is not fully available, the collected information about the environment is to be taken into account. This stems from the fact that the trajectory determines the amount of data that can be perceived. The perceptive information may include the point of view of the onboard camera, the obstacles (situational awareness), or the texture richness of the environment. In addition to the trajectory planning, one significant task is the trajectory tracking control, i.e., the design of the control commands to make the MRS stable as well as to track the trajectory as close as possible. The tracking task satisfies the desired position, velocity and orientation while fulfilling the given task. Figure 1 provides an overview of the architecture of the proposed solution.

Generally, the component includes the following parts, which are summarized in Figure 2. In this figure, the features of the components are included. In fact, Figure 2 provides a summary of how the points mentioned in Section 1 are addressed in the proposed architecture, presented in Figure 1 which includes the following parts

- Higher level centralized planner: it receives the overall task plan and objectives and decomposes it into different subtasks for MRS members, as sets of waypoints, assigned to each member.
- Lower-level decentralized planner: it receives the waypoints and plans a trajectory for the corresponding member, taking into account, model parameters, states, situational information, security, safety and quality assurance metrics.
- Lower level decentralized control: it receives the planned trajectory and sends the corresponding commands to MRS to follow that trajectory.
- An algorithm such that non-attacked robots will generate a rescue trajectory providing further sensor feedback to the compromised. It should be noted that this component is going to be embedded into the lower-level decentralized planner and lower level decentralized control. This will be tackled as part of D2.5: Multi-Robot Collaborative Rescue Mission (M30).

The trajectory planning and tracking to tackle the navigation task, can be generally seen as Algorithm 1, where $x_d(t)$ is the planned trajectory, $J(t)$ is an optimization function, and $u_d(t)$ is the designed control.

The Algorithm 1, is to be implemented and solved at the lower-level decentralized planner. In this algorithm, we design the desired trajectory $x_d(t)$, as a solution to minimization of $J(t)$. Then, at the lower-level decentralized tracker the control command $u(t)$ is designed to steer the robot states $x(t)$ towards $x_d(t)$. Moreover, the stability of the closed-loop system is guaranteed.

3.2 Higher level centralized planner

Higher level centralized planner receives the overall task plan and objectives and decomposes it into different subtasks for MRS members, as sets of waypoints, assigned to each member. This planner is to be offline for

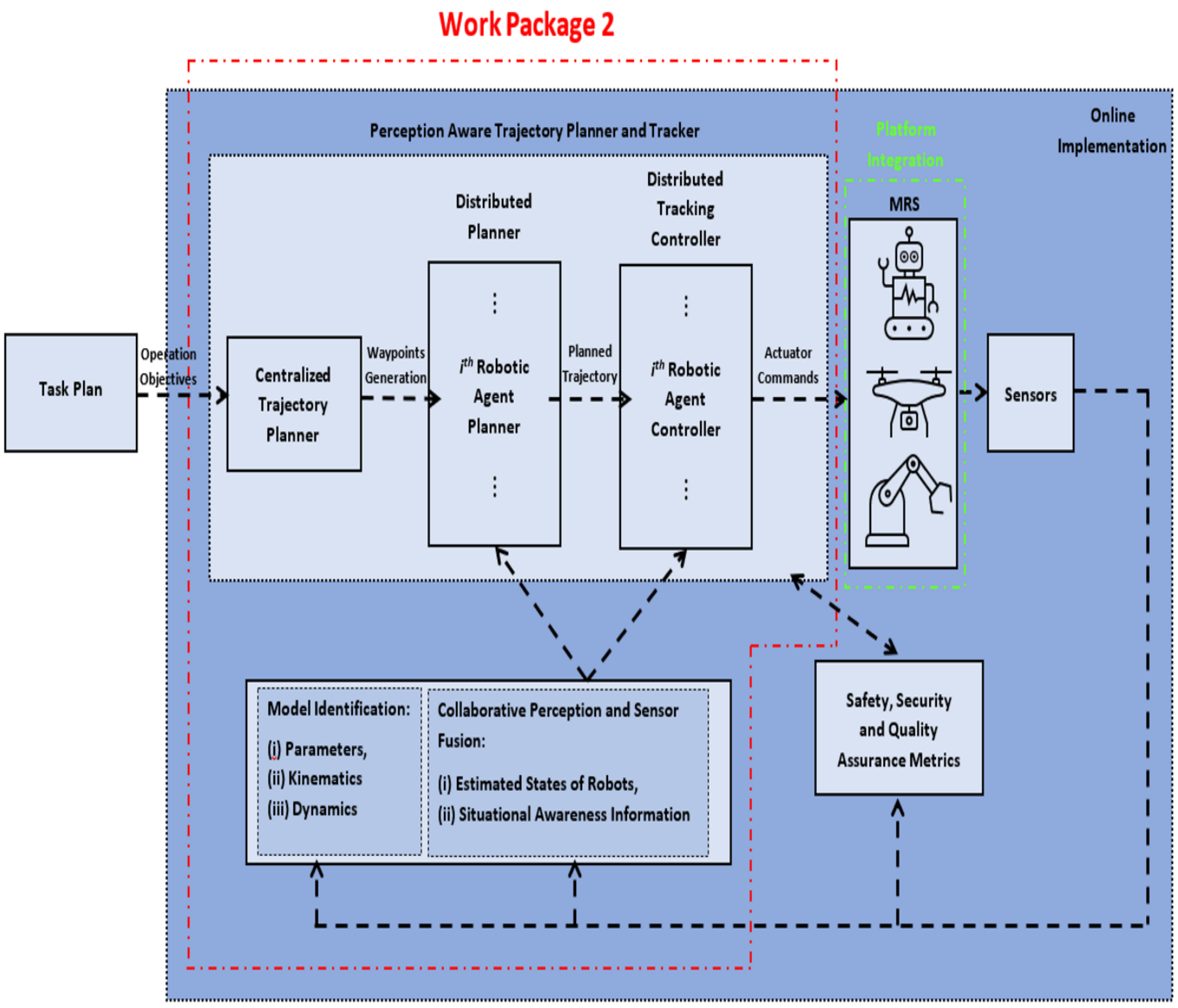
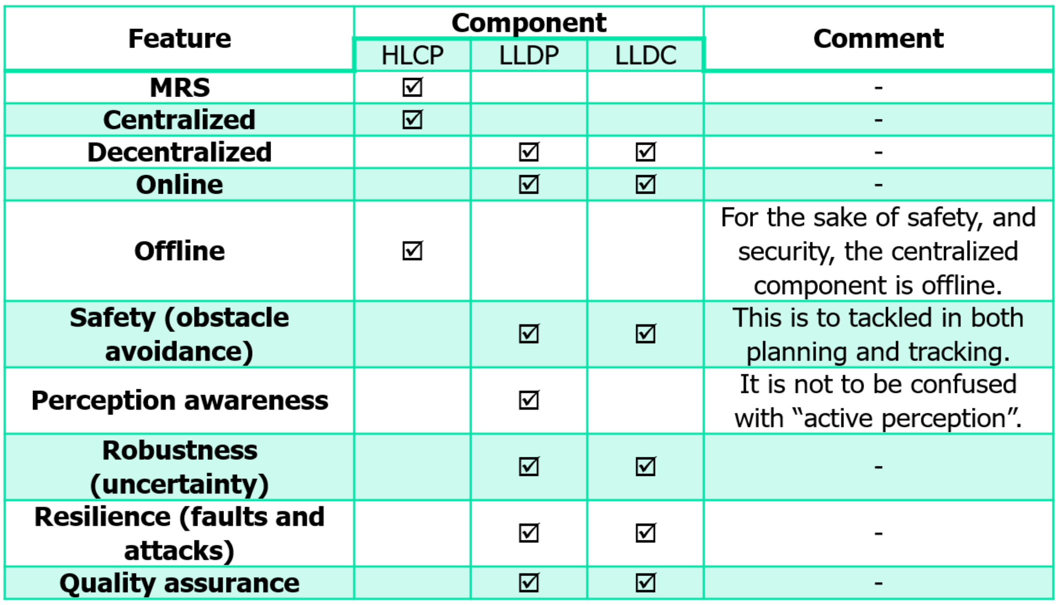

Work Package 2
Perception Aware Trajectory Planner and Tracker
Platform Integration
Online Implementation
Distributed Planner
Distributed Tracking Controller
MRS
Task Plan
Operation Objectives
Centralized Trajectory Planner
Waypoints Generation
i^{th} Robotic Agent Planner
Planned Trajectory
i^{th} Robotic Agent Controller
Actuator Commands
Sensors
Model Identification: (i) Parameters, (ii) Kinematics (iii) Dynamics
Collaborative Perception and Sensor Fusion: (i) Estimated States of Robots, (ii) Situational Awareness Information
Safety, Security and Quality Assurance Metrics

Figure 1: Overall structure of trajectory planning and tracking component

Algorithm 1 Perception aware MRS trajectory planning and tracking

```
1: Design $x_d(t)$, by optimizing $J(t)$, subject to
     • Passing through the waypoints (defined based on the decomposed task),
     • Identified dynamics of the robot and estimated states,
     • Actuator limit,
     • Keeping distance by the estimated obstacle position,
     • Retaining in /avoiding from some areas or points (perceptive information),
     • Satisfying the safety and security metrics.
2: Design $u(t)$, by taking into account the current estimated states of the robot and designed $x_d(t)$,
```



Feature	Component			Comment
	HLCP	LLDP	LLDC	
MRS	☑			-
Centralized	☑			-
Decentralized		☑	☑	-
Online		☑	☑	-
Offline	☑			For the sake of safety, and security, the centralized component is offline.
Safety (obstacle avoidance)		☑	☑	This is to tackled in both planning and tracking.
Perception awareness		☑		It is not to be confused with "active perception".
Robustness (uncertainty)		☑	☑	-
Resilience (faults and attacks)		☑	☑	-
Quality assurance		☑	☑	-

Figure 2: Trajectory planning and tracking component features. HLCP: Higher level centralized planner, LLDP: Lower-level decentralized, planner, LLDC: Lower-level decentralized control

the sake of the security of the overall task plan. It also coincides with the ExSce as well as the task plan. Given the different natures of the use cases and different operational scenarios, this can be foreseen either in the Specification of MRS Capabilities or the ExSce. For instance, for vineyard spraying operations, in the task plan, it has already implied the plants to be sprayed, either offline or by another drone that is covering the whole vineyard to detect the plants. Therefore, the higher level centralized planner indeed can be determined depending on the nature of the operation. however, we have placed this planner in the architecture Figure 1, to be compatible with the different use cases in a generic manner. In fact, this planner will provide the waypoints to the lower-level decentralized planner. Therefore, given this architecture, different operational scenarios for different use cases can be tackled. Therefore, without loss of generality, we assume that this higher-level planner has been already foreseen in the task plan or ExSce, providing the waypoints.

3.3 Lower-level decentralized planner

From the higher level planner, the waypoints W_i of each robotic agent, for $i \in \{1, ..., K\}$, are determined and sent to the lower level decentralized planner, where the trajectory planning for each robot is obtained as the solution to an optimization problem. It is guaranteed that the robotic agent passes through the corresponding waypoints. The trajectory planning is mathematically formulated as

$$x_d(t) = \underset{x(t)}{\operatorname{argmin}}\, J\left(x(t)\right), \tag{1}$$

subject to

$$\begin{aligned} J\left(x(t)\right) &= t_f\left(x_f, x(t)\right) - t_0, \\ x(t_0) &= x_0, \\ x(t_f) &= x_f, \\ x(t_i) &= W_i, \\ \dot{x}(t) &= f(x(t), u(t)), \\ y(t) &= x(t) + \epsilon_x(t), \\ p(t) &= P(x(t), P^*) \\ x_L \leq x(t) &\leq x_U, \\ u_L \leq u(t) &\leq u_U, \\ p_L \leq p(t) &\leq p_U, \\ R_k &\leq \left\| x(t) - x_{obs,k}(t) \right\|, \end{aligned} \tag{2}$$

where, $x(t)$ is the dynamics state vector, t_f and t_0 are final and initial operation times, respectively, x_f and x_0 are final and initial states, respectively, $u(t)$ is the control command, $t_0 \leq t_i \leq t_f$, for $i \in \{1, ..., K\}$, are increasing time sequence, W_i is the state of the i^{th} waypoint, $f(x(t), u(t))$ is the dynamics equations governing the motion of the robot, $y(t)$ is the measurements vector with a noise vector $\epsilon_x(t)$, and $p(t)$ is the perceptive index governed by the equation $P(x(t), P^*)$ with respect to the point/area of interest P^*. X_L and X_U denote the element-wise lower and upper bound vectors on the variable vector $X(t)$. Finally, $x_{obs,k}(t)$ and R_k represent the position and safety radius of k^{th} obstacle. In fact, $R_k \leq \|x(t) - x_{obs,k}(t)\|$ imposes the constraint to keep the state of the robot outside of the safety sphere around the obstacle.

To solve the optimization (1), there are many approaches, as reviewed in Section 2. However, as we analyzed the existing works, the main issue with the real-time implementation and fast optimization is the implementation of the constraints (2). Specifically, the inequality constraints might cause issues in the convergence of the solver. Accordingly, it is really crucial how we implement the inequality constraints. As discussed later, we are going to use the Legendre pseudospectral method to transcribe the continuous optimization problem into an equivalent discrete one. The trajectory is represented by a number of Legendre functions passing through a number of collocation points. Then, the position of these collocation points is as the optimization variables. Therefore, the constraints are to be applied on these collocation points.

To apply the obstacle avoidance constraint $R_k \leq \|x(t) - x_{obs,k}(t)\|$ to the collocation points, the common condition to be checked is the distance of the points from the obstacle to be greater than the safety radius. However, this still can lead to the safety violation as illustrated in Figure 3, for two collocation points $C_{j,1}$ and $C_{j,2}$.

As it is shown in Figure 3, the collocation points $C_{j,1}$ and $C_{j,2}$ are outside of the safety sphere, i.e., $r_{i,j,1} > R_i$ and $r_{i,j,2} > R_i$. However, the line segment $\overline{C_{j,1}C_{j,2}}$ is still passing through the safety sphere. One obvious solution is to increase the number of collocation points and apply the distance constraint. However, this increases the computational burden of the problem. More importantly, it is still not guaranteed that the connecting lines are moved outside, as illustrated in Figure 4, where $\overline{C_{j,2}C_{j,3}}$ passes through the safety sphere.

The resolution to this is to consider the perpendicular distance for each line segment from the obstacle, i.e., $d_{i,j,k}$, for $k = 0, \ldots, N$, where N represents the number of collocation points, $C_{j,0} = w_j$ and $C_{j,N+1} = w_{j+1}$, as illustrated in Figure 5.

It should be noted that in Figure 5, $d_{i,j,k}$, for $k = 0, \ldots, 4$, are all same for all the segments, as they are on the same straight line. Considering only perpendicular distance as the constraint $d_{i,j,k} > R_i$, will lead to a very conservative approach, as it will move the points $C_{j,1}$ and $C_{j,4}$ to some place to satisfy the constraint, as shown in Figure 6.

This takes a lot of workspaces and might lead to an infeasible problem after applying the other constraints. Therefore, we only need the constraint to be applied to $C_{j,2}$ and $C_{j,3}$ in Figure 5, as they lie within the safety

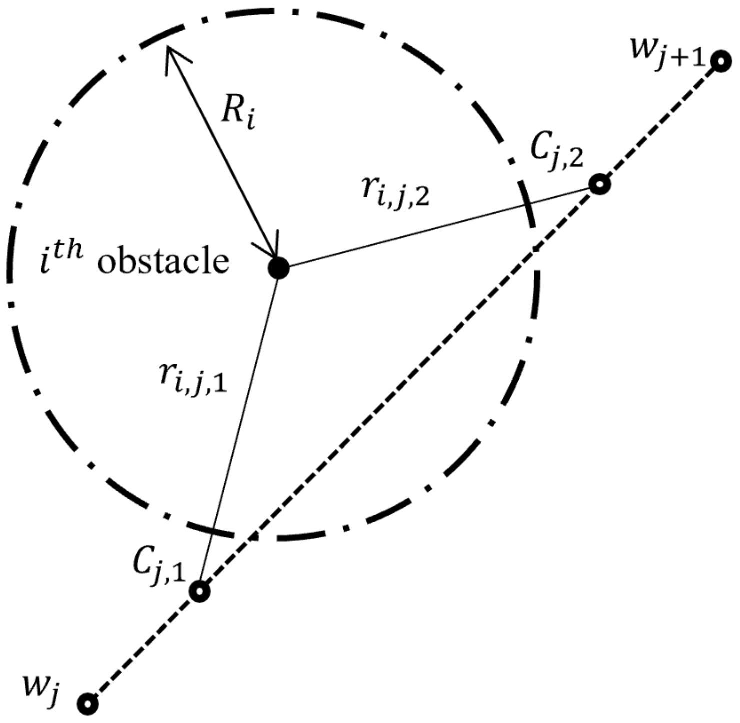


Figure 3: Violation of safety region for the case the waypoints and collocation points are still outside.

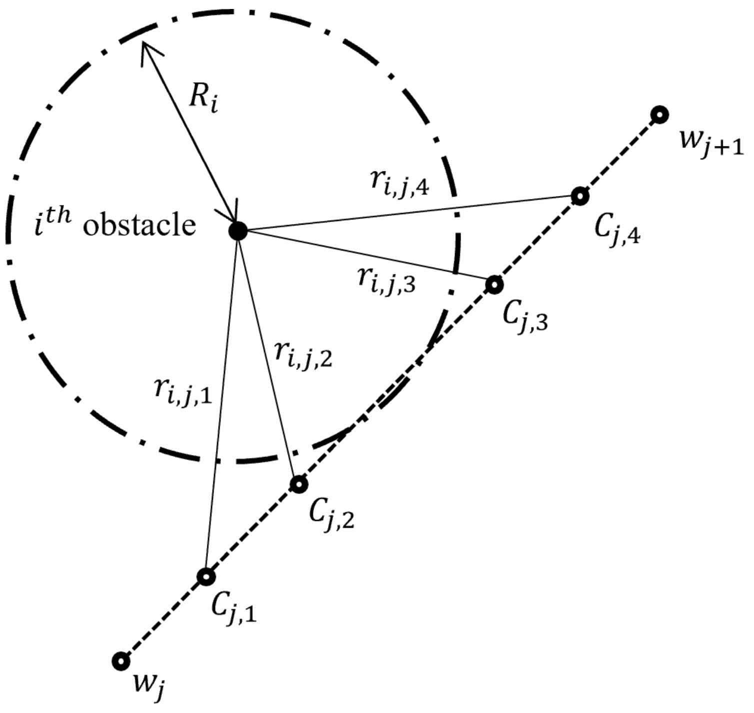


Figure 4: Violation of safety region with an increased number of collocation points.

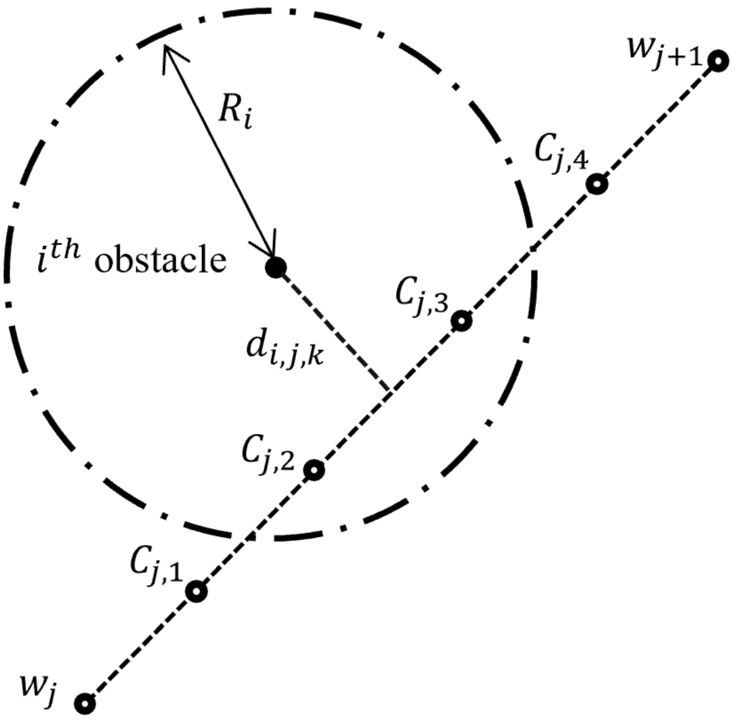


Figure 5: Perpendicular distance check.

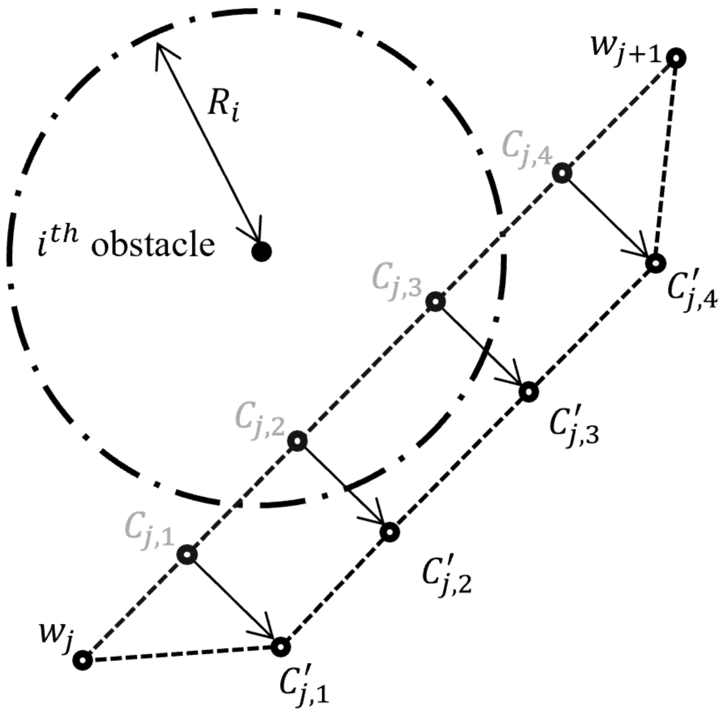


Figure 6: Applying the perpendicular distance check to all the points. The points with ($'$) represent the new position of the points in grey.

sphere. The solution we propose is to first check if the projection of i^{th} obstacle position on the line segment $\overline{C_{j,k}C_{j,k+1}}$ lies on the segment or its extension. To do so, we compute

$$t_{i,j,k} = \frac{(O_i - C_{j,k}) \,.\, (C_{j,k+1} - C_{j,k})}{(C_{j,k+1} - C_{j,k}) \,.\, (C_{j,k+1} - C_{j,k})}, \tag{3}$$

where, O_i is the position of i^{th} obstacle and $\cdot$ represents the inner product. Then, if $0 < t_{i,j,k} < 1$, then the projection point is on the line segment $\overline{C_{j,k}C_{j,k+1}}$. Otherwise, it is on its extension. This is illustrated in Figure 7, where, $t_{i,j,k} < 0$, $0 < t_{i,j,k+1} < 1$ and $1 < t_{i,j,k+2}$. Then if $0 < t_{i,j,k} < 1$ and $d_{i,j,k} < R_i$, then the line segment $\overline{C_{j,k}C_{j,k+1}}$ is passing through the safety sphere. In this case, we apply the constraint in the optimization to move the $\overline{C_{j,k}C_{j,k+1}}$ outside of the safety sphere. An example is illustrated in Figure 8.

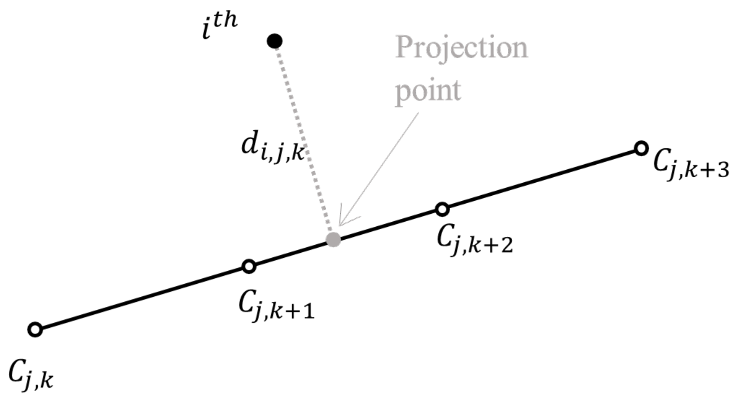


Figure 7: Projection point check.

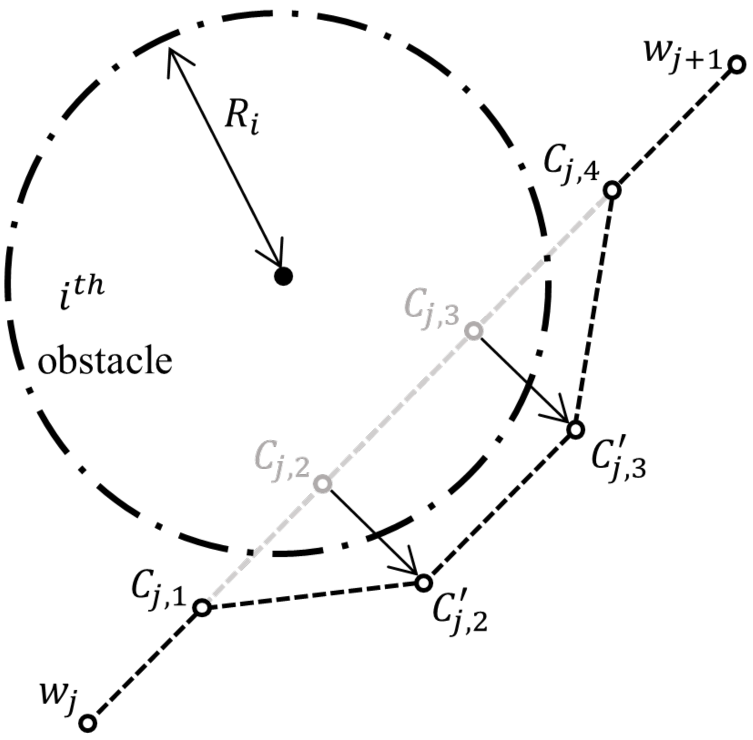


Figure 8: Applying the projection point check.

In summary, this condition is as follows.

```
**if** $0 < t_{i,j,k} < 1 \,\&\, d_{i,j,k} < R_i$ **then**
    move the $C_{j,k}$ , for $k = 1, \ldots, N$, such that $\overline{C_{j,l}C_{j,l+1}}$, for $l = 0, \ldots, N$ is outside of the safety sphere.
**end if**
```

This approach theoretically solves the mentioned problem, however, it imposes an issue on the implementation of the optimization problem. This stems from the presence of the *if* condition in the construction of constraints which is a common approach for obstacle avoidance constraints [30]. Using *if* condition changes the structure and size of the constraints at every iteration of the solver. This is due to the fact that by moving the collocation points, the constraint might be satisfied and it will be removed from the constraints matrix. More importantly, *if* condition is not a smooth condition which might make a problem with the convergence of the solver. To resolve this problem, we define the following smooth constraint avoiding the use of *if* condition.

$$f_{i,j,k} := \frac{1}{2}\Gamma_{i,j,k}\left(R_i - d_{i,j,k}\right) \leq 0, \tag{4}$$

where $\Gamma_{i,j,k} = \tanh\left(\frac{t_{i,j,k}}{\delta}\right) - \tanh\left(\frac{t_{i,j,k}-1}{\delta}\right)$ and $\delta > 0$ is a small scalar. It is readily shown that for $0 < t_{i,j,k} < 1$, $\Gamma_{i,j,k} \approx 1$. Therefore, $f_{i,j,k} = R_i - d_{i,j}$. Then the constraint $f_{i,j,k} \leq 0$ is equivalent to $R_i - d_{i,j,k} \leq 0$. On the other hand, for $t_{i,j,k} \leq 0$ or $1 \leq t_{i,j,k}$, the term $\Gamma_{i,j,k} = 0$ and, hence, $f_{i,j,k} = 0$. Moreover, the condition $0 \leq 0$ is already satisfied and it does not apply the distance constraint on the points $C_{j,k}$ and $C_{j,k+1}$. So, this constraint encapsulates both conditions for the line segment on which the projection point lies, without using if condition. More importantly, this constraint is smooth at points $t_{i,j,k} = 0$ and $t_{i,j,k} = 1$. $f_{i,j,k}$ is illustrated in the following figure for $\delta = 0.05$ and $R_i = 1.5$.

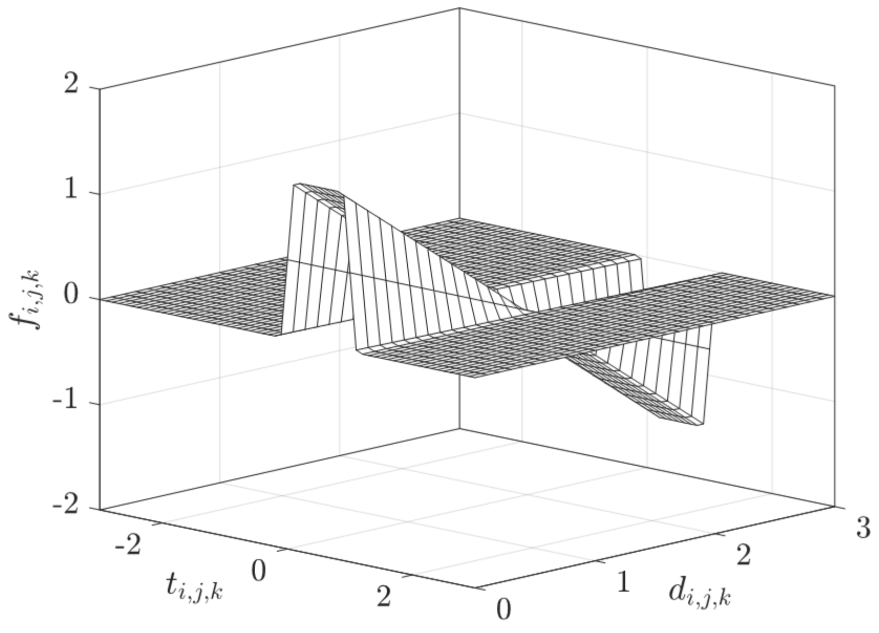


Figure 9: Smooth obstacle avoidance constraint.

Another improvement to make the optimization problem feasible in real-time settings with fast convergence of the solver is considering the initial guess. For the cases, the initial guess goes through the safety bond. This might make the optimization problem infeasible. So, we have to avoid this. For this case, we first radially move the collocation points out of the sphere. To do so, for each point, we check if it is inside of the sphere. if so, we move the point in the direction of the carrying radius, i.e., connecting the centre to the point. Therefore, it is guaranteed that the collocation points are placed outside. However, this does not necessarily guarantee that the connecting line does not pass through the sphere. for this aim, we can consider the perpendicular distance of the centre to each line. However, just moving all the points according to the perpendicular distance might make lead to a conservative initial guess which is away from the optimal solution. So, we only move the lines, on which the projection point lies, not on their extension. If the projection point lies on the line and if the perpendicular distance is smaller than the radius, then we move the line in the perpendicular direction to make sure the line lies outside of the sphere. This improvement might make the initial guess non-smooth and longer than the direct connecting line. So, finally, we shorten the guess, i.e., we check if the direct line between the collocation points pas through the sphere. if not, the direct line is replaced. In Figure 10, these improvements are illustrated.

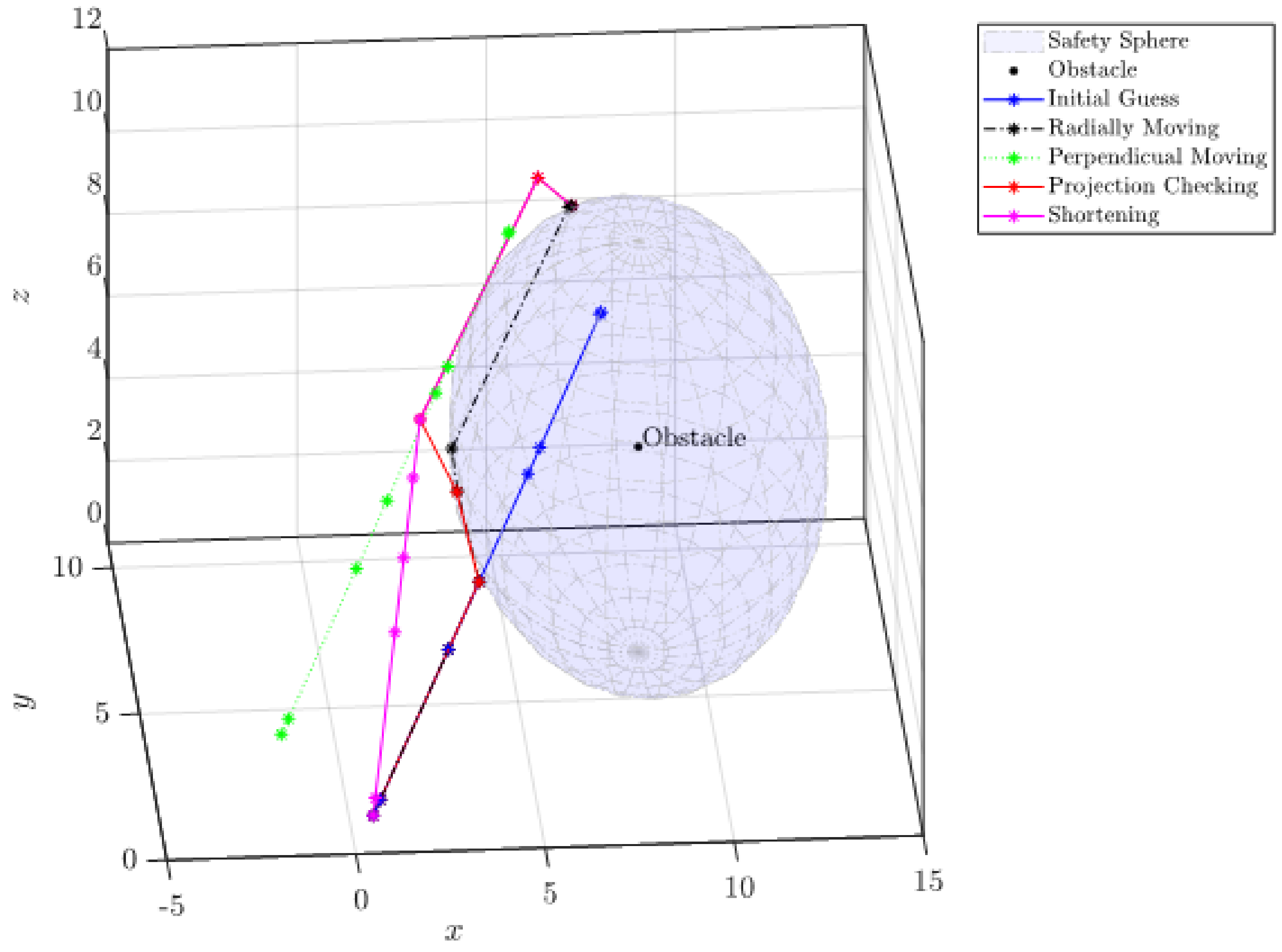


Figure 10: Initial guess improvement.

Now, to solve the continuous-time optimization problem (2), it is transcribed into a nonlinear programming (NLP) problem using Legendre pseudospectral method [73]. To go through the transcription process, let us consider a trajectory segment between waypoints W_k and W_{k+1}. The state and control histories between these waypoints are approximated by Lagrange polynomials in time, illustrated in Figure 11, as

$$x^k(\tau) \approx \sum_{i=1}^{n} x_i^k \phi_i(\tau), \tag{5a}$$

$$u^k(\tau) \approx \sum_{i=1}^{n} u_i^k \phi_i(\tau), \tag{5b}$$

where τ is time, x_i and u_i are state and control approximation vectors at nodes, k denotes segment number and i denotes node number, ϕ_i is the basis function given by

$$\phi_i(\tau) = \prod_{j=1,j\neq i}^{n} \frac{\tau - \tau_j}{\tau_i - \tau_j}, \tag{6}$$

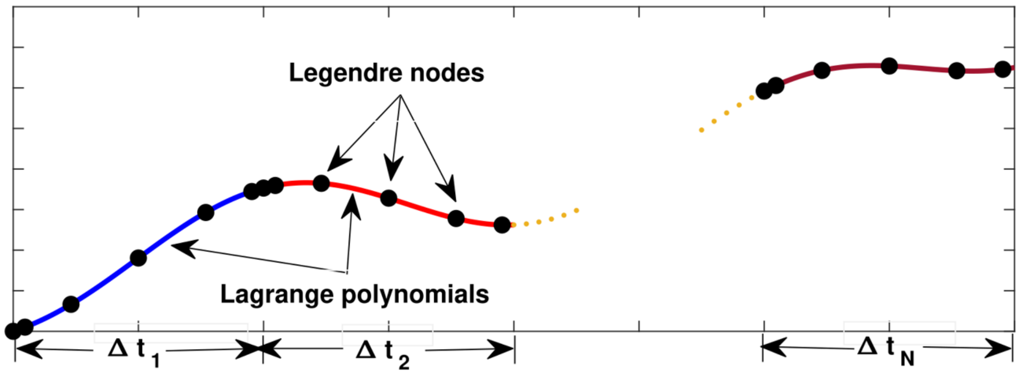


Figure 11: Illustration of discretization in Legendre pseudospectral method.

for which

$$\phi_i(\tau_j) = \begin{cases} 0, & \text{if } i = j, \\ 1, & \text{otherwise.} \end{cases} \tag{7}$$

Times τ_i are defined as

$$\tau_i = \frac{1 + \tilde{\tau}_i}{2}, \tag{8}$$

where $\tilde{\tau}$ are -1, 1 and the roots of Legendre polynomial of order $n-2$

$$\frac{1}{2^{n-2}(n-2)!} \frac{d^{n-2}}{dp^{n-2}} \left(p^2 - 1\right)^{n-2} = 0. \tag{9}$$

Let t_k and t_{k+1} be the times at which dynamics is at waypoints W_k and W_{k+1}, respectively. Let $\Delta^k t$ be the time-interval of this segment. The actual and non-dimensionalized times are related as

$$\tau = \frac{t - t_k}{\Delta^k t}, \tag{10}$$

$$d\tau = \frac{dt}{\Delta^k t}. \tag{11}$$

The Lagrange polynomials are collocated at the roots of Legendre polynomials to result in a set of algebraic equations

$$\sum_{i=1}^{n} x_i^k \dot{\phi}_i(\tau) = \dot{x}^k(\tau). \tag{12}$$

This can be rewritten as

$$\sum_{i=1}^{n} x_i^k \dot{\phi}_i(\tau) = \Delta^k t f(x_i^k, u_i^k). \tag{13}$$

To ensure that trajectories are smooth, connectivity and continuity constraints are imposed at the terminal and initial ends of consecutive segments

$$x_n^k = x_0^{k+1}, \tag{14}$$

$$u_n^k = u_0^{k+1}, \tag{15}$$

$$\sum_{i=1}^{n} x_i^k \dot{\phi}_i(1)/\Delta^k t = \sum_{i=1}^{n} x_i^{k+1} \dot{\phi}_i(0)/\Delta^{k+1} t, \tag{16}$$

$$\sum_{i=1}^{n} u_i^k \dot{\phi}_i(1)/\Delta^k t = \sum_{i=1}^{n} u_i^{k+1} \dot{\phi}_i(0)/\Delta^{k+1} t. \tag{17}$$

The cost function J, defined in (1), is then reduced to the sum of time intervals of each segment, i.e.,

$$J = \sum_{k=1}^{N} \Delta^k t. \tag{18}$$

The cost function in (18), constraints in (2) and (4), constitute the NLP problem. After the transcription, the problem is solved using interior point algorithm [74] implemented in IPOPT solver [75] interfaced with Matlab. Moreover, CasADi is used for parametric auto differentiation of cost function and the constraints, for speeding up the optimization solution. The over architecture is shown in Figure 12.

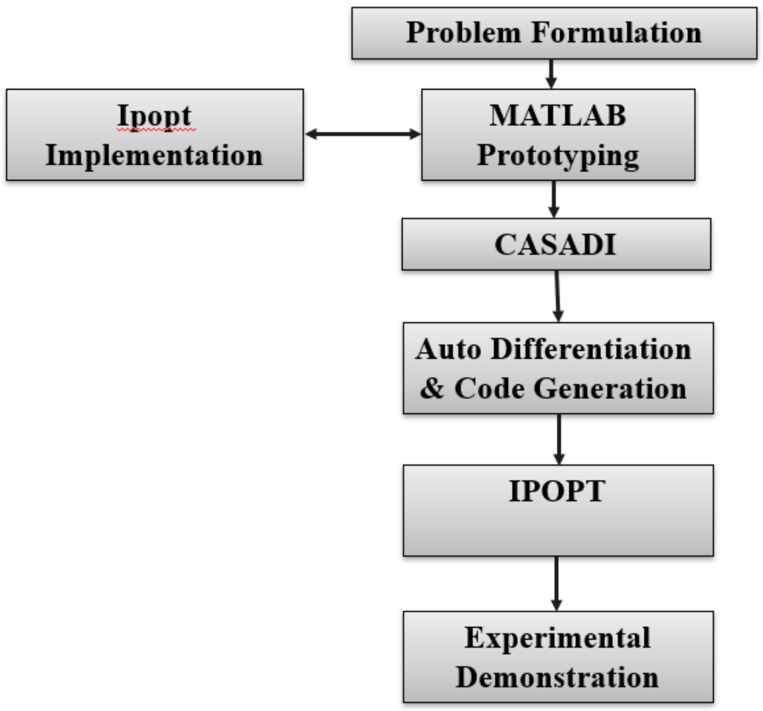


Figure 12: Architecture of the optimization problem for trajectory planning.

In Figure 13, an initial comparison of optimization time for trajectory planning of drone dynamics is given. It is obvious that by the implementation of the proposed approach the optimization time is considerably reduced.

3.4 Lower level decentralized control

After solving the optimization (1) with constraints (2), in the tracking control part, we design $u(t)$ such that

$$\lim_{t \to t_c} \|x(t) - x_d(t)\| \leq \epsilon_c, \tag{19}$$

for any x_0, with convergence time t_c and convergence radius ϵ_c. Ideally, ϵ_c is going to be zero.

In this section, we design a controller for accurate trajectory tracking with disturbances and unknown uncertainties in the dynamics. We use an Adaptive Super-Twisting Controller (ASTC) which is robust and gain adaptation minimizes chattering with no information required on disturbance bounds.

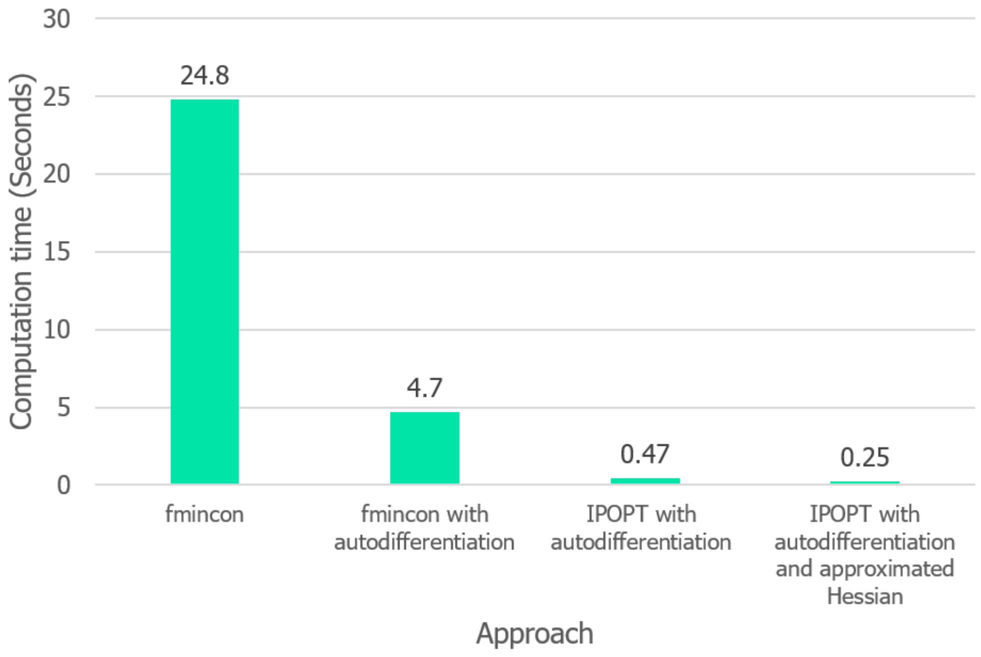


Figure 13: Comparison of computation time using different approaches.

It should be noted that the ASTC is designed for dynamics of quadrotor UAV in this section. However, the proposed approach can be simply applied for different robot types. The kinematics of the quadrotor UAV can be presented as

$$\dot{\underline{x}}\,(t) = \underline{v}\,(t)\,, \tag{20a}$$

$$\dot{\underline{v}}(t) = \frac{1}{M}\left(R(t)\underline{F}(t) - M\underline{F}_g + \underline{f}_d(\underline{v}, \underline{\Phi}, t)\right), \tag{20b}$$

$$\dot{\underline{\Phi}}(t) = R_q^{-1}(t)\underline{\omega}(t), \tag{20c}$$

$$\dot{\underline{\omega}}(t) = J^{-1}\left(\underline{\tau}(t) - \underline{\omega}(t) \times J\underline{\omega}(t) + \underline{\tau}_d(\underline{\omega}, t)\right), \tag{20d}$$

where, $\underline{x}(t) = [x(t), y(t), z(t)]^T \in \mathbb{R}^3$ and $\underline{v}(t) = [v_x(t), v_y(t), v_z(t)]^T \in \mathbb{R}^3$ are position and velocity vectors of the centre of gravity, respectively, in the inertial frame. $\underline{\omega}(t) \in \mathbb{R}^3$ is the angular velocity in the body frame. M is the known mass and $J \in \mathbb{R}^{3\times3}$ is the unknown mass moment inertia. $\underline{\Phi}(t) = [\theta(t), \phi(t), \psi(t)]^T \in \mathbb{R}^3$ is the Euler angle vector and $\underline{F}(t) = [0, 0, f(t)]^T \in \mathbb{R}^3$, both in the the body frame. $f(t) \in \mathbb{R}_+$ is the thrust. $\underline{F}_g = [0, 0, g]^T \in \mathbb{R}^3$ is the vector of gravity force in the inertial frame. $\underline{\tau}(t) \in \mathbb{R}^3$ is the torques vector in the body frame. Also, $\underline{f}_d(\cdot) \in \mathbb{R}^3$ and $\underline{\tau}_d(\cdot) \in \mathbb{R}^3$ are unknown disturbances of translational and rotational dynamics, respectively. The rotation matrices $R(t) \in \mathsf{SO}(3)$ and $R_q(t) \in \mathsf{SO}(3)$ are defined as [76]

$$R(t) = \begin{bmatrix} C_\psi C_\theta & C_\psi S_\theta S_\phi - S_\psi C_\phi & C_\psi S_\theta C_\phi + S_\psi S_\phi \\ S_\psi C_\theta & S_\psi S_\theta S_\phi + C_\psi C_\phi & S_\psi S_\theta C_\phi - C_\psi S_\phi \\ -S_\theta & C_\theta S_\phi & C_\theta C_\phi \end{bmatrix}, \tag{21a}$$

$$R_q(t) = \begin{bmatrix} 1 & 0 & -S_\theta \\ 0 & C_\phi & C_\theta S_\phi \\ 0 & -S_\phi & C_\theta C_\phi \end{bmatrix}, \tag{21b}$$

where, C_α and S_α denote cosine and sine of α, respectively. For the sake of notation simplicity, we present the translational dynamics, i.e., (20b), as

$$\dot{\underline{v}}(t) = a\underline{u}(t) - \underline{F}_g + \underline{f}_1(\underline{v}, \underline{\Phi}, t), \tag{22}$$

where, $u(t) = R(t)\underline{F}(t) = [u_x(t), u_y(t), u_z(t)]^T \in \mathbb{R}^3$, $a = 1/M$, and $\underline{f}_1(\cdot) = [f_{1,x}(\cdot), f_{1,y}(\cdot), f_{1,z}(\cdot)]^T \in \mathbb{R}^3 = a\underline{f}_d(\cdot)$ is lumped disturbance vector. It is obvious that the dynamics (22) can represent a variety of robotic dynamics. Therefore, we design the ASTC considering the dynamics (22).

Considering (22), we aim to design the desired thrust $f(t)$, desired Euler angle $\underline{\Phi}_d(t) = [\theta_d(t), \phi_d(t), \psi_d(t)]^T \in \mathbb{R}^3$ and desired angular velocity $\underline{\omega}_d(t)$, such that to steer the UAV position $\underline{x}(t)$ towards the known desired position $\underline{x}_d(t) = [x_d(t), y_d(t), z_d(t)]^T \in \mathbb{R}^3$, for any initial conditions $\underline{x}(0)$. For this, we design $u(t)$ in (22), and accordingly, the desired thrust $f(t)$ and orientation $\Phi_d(t)$ are obtained. It is worth noting that $\underline{x}_d(t)$ has been designed in the previous section.

Remark 1. *It should noted that the proposed controller can be readily extended for the attitude dynamics. However, the designed offboard controller can only run at maximum frequency of 50-100 Hz. On the other hand, the desired torque $\underline{\tau}(t)$ and the corresponding motor Pulse Width Modulation (PWM) signals need to be computed at a high rate, e.g., 1000 Hz. Due to this computational limitation, these control inputs are left to be computed by the available onboard controller.*

Assumption 1. *It is assumed that the UAV's states $\underline{x}$, $\underline{v}$, and $\underline{\Phi}$ are available for control design [77]. Furthermore, it is assumed that the time derivative of $f_{1,i}$ is bounded as $|\dot{f}_{1,i}(\cdot)| \leq a_{1,i}$ and $|\ddot{f}_{1,i}(\cdot)| \leq a_{2,i}$ for $i \in \{x, y, z\}$, where $a_{1,i}$ and $a_{2,i}$ are positive constants.*

Remark 2. *In Assumption 1, the UAV's states are assumed to be known, by on-board sensors, inertial measurement unit, extended Kalman filter or a motion capture system [78]. However, the measurement errors can be encapsulated in the lumped disturbance in* (20). *This further motivates the use of ASTC for compensating for the effects of these errors. Also, given the nature of external disturbance, it is reasonable to assume its time derivatives are bounded [79]. This is a relaxed condition, in comparison to the assumption of bounded disturbance norm, used in some works [80–82].*

The following lemmas are used in the design procedure.

Lemma 1. *Consider the dynamics*

$$\dot{x}_1(t) = -\lambda|x_1(t)|^{0.5} sign\left(x_1(t)\right) + x_2(t) + d(t), \tag{23a}$$
$$\dot{x}_2(t) = -k(t) sign\left(x_1(t)\right). \tag{23b}$$

Under the assumption that $|\dot{d}| \leq d_1$ and $|\ddot{d}| \leq d_2$, where d_1 and d_2 are positive constants. For any given $x_1(0)$, there exists a sufficiently large $\lambda \geq 0$, such that, provided $k(t) \geq d_1$, both x_1 and $\dot{x}_1$ converge to zero in finite time, i.e., it induces a second order sliding mode, and $w_{eq}(t) = \dot{d}(t)$, where $w_{eq}(t)$ is the low-pass filtered version of $w(t) = k(t) sign\left(x_1(t)\right)$ [83]. Furthermore, the solution of (23) *needs to be understood in a Filippov sense.*

Lemma 2. *Considering the system $\dot{x}(t) = f(x, u)$ with the state vector $x(t)$ and the control input $u(t)$, and assume $V\left(x(t)\right)$ as a smooth positive definite function, satisfying $\dot{V}(\cdot) \leq -\alpha\sqrt{V(\cdot)}$, along the trajectories of $x(t)$. Then, for any initial condition $x(0)$, the trajectory $x(t)$ is stable and converges to zero in finite time smaller than $T\left(x(0)\right) = 2\sqrt{V\left(x(0)\right)}/\alpha$ [84].*

Lemma 3. *For all $x, y \in \mathbb{R}_+$, the inequality $x + y \geq \left(x^2 + y^2\right)^{0.5}$ holds.*

To construct the control, we define the sliding surfaces as

$$\sigma_x(t) = (v_x(t) - v_{d,x}(t)) + \lambda_x(x(t) - x_d(t)), \tag{24a}$$
$$\sigma_y(t) = (v_y(t) - v_{d,y}(t)) + \lambda_y(y(t) - y_d(t)), \tag{24b}$$
$$\sigma_z(t) = (v_z(t) - v_{d,z}(t)) + \lambda_z(z(t) - z_d(t)), \tag{24c}$$

where $\underline{v}_d(t) = [v_{d,x}(t), v_{d,y}(t), v_{d,z}(t)]^T = \dot{\underline{x}}_d(t) \in \mathbb{R}^3$ is desired linear velocity vector. λ_x, λ_y and $\lambda_z \in \mathbb{R}_+$ are design parameters. Now, we design the control $u(t)$ in (22) for $i \in \{x, y, z\}$ as

$$u_i = \frac{-k_{1,i}|\sigma_i(t)|^{0.5} sign(\sigma_i(t)) + \hat{w}_i(t) + f_i(t)}{a}, \tag{25a}$$

$$\dot{\hat{w}}_i(t) = -k_{2,i}(t) sign(\sigma_i(t)), \tag{25b}$$

where $k_{1,i} \in \mathbb{R}_+$ is the design parameters, and

$$f_x(t) = a_{d,x}(t) - \lambda_x \left(v_x(t) - v_{d,x}(t)\right), \tag{26a}$$

$$f_y(t) = a_{d,y}(t) - \lambda_y \left(v_y(t) - v_{d,y}(t)\right), \tag{26b}$$

$$f_z(t) = a_{d,z}(t) - \lambda_z \left(v_z(t) - v_{d,z}(t)\right) + g, \tag{26c}$$

where $\underline{a}_d(t) = [a_{d,x}(t), a_{d,y}(t), a_{d,z}(t)]^T = \ddot{\underline{x}}_d(t) \in \mathbb{R}^3$ is desired linear acceleration vector, with the adaptive laws

$$\dot{\hat{w}}_{i,eq}(t) = \frac{1}{\tau_i}\hat{w}_i(t) - \frac{1}{\tau_i}\hat{w}_{i,eq}(t), \tag{27a}$$

$$\dot{r}_i(t) = \gamma_i|\delta_i(t)| + r_{0,i}\sqrt{\gamma_i} sign(e_i(t)), \tag{27b}$$

$$\dot{k}_{2,i}(t) = -(r_{0,i} + r_i(t)) sign(\delta_i(t)), \tag{27c}$$

where $0 < \tau_i < 1$ is constant, and $r_{0,i}, \gamma_i \in \mathbb{R}_+$ are design parameters. $\delta_i(t)$ and $e_i(t)$ in (27b) and (27c) are defined as

$$\delta_i(t) = k_{2,i}(t) - \frac{1}{\alpha_i}|\hat{w}_{i,eq}(t)| - \epsilon_i, \tag{28a}$$

$$e_i(t) = \frac{q_i}{\alpha_i}a_{2,i} - r_i(t), \tag{28b}$$

respectively, where $0 < \alpha_i < 1$ is selected sufficiently small and $\epsilon_i \in \mathbb{R}_+$ is selected sufficiently large design parameters, and $q_i > 1$ to ensure $|d\hat{w}_{i,eq}(t)/dt| \leq q_i a_{2,i}$.

Remark 3. *Considering* (27a)*, for* $\hat{w}_{i,eq}(0) = 0$*, it is easy to obtain* $\hat{w}_{i,eq}(s) = \frac{1}{\tau_i s+1}\hat{w}_i(s)$*, in frequency domain. The transfer function* $\frac{1}{\tau_i s+1}$ *represents a low-pass filter for small* τ_i*. Therefore,* $\hat{w}_{i,eq}(t)$ *is the low-pass filtered version of* $\hat{w}_i(t)$*. Moreover, as studied in [85], these exist constants* $0 < \eta_1 < 0$ *and* $0 < \eta_2$ *satisfying* $|\hat{w}_i(t) - \hat{w}_{i,eq}(t)| \leq \eta_1|\hat{w}_i(t)| + \eta_2$*. Also, there exists* q_i *satisfying* $|d\hat{w}_{i,eq}(t)/dt| \leq q_i a_{2,i}$ *[86].*

Remark 4. *Regarding design parameters, the following points are worth noting. In* (24)*,* λ_i *induces the exponential convergence rate. Therefore,* λ_i *is to be selected carefully, to avoid jerky motion of UAV. In* (25a)*,* $k_{1,i}$ *is selected sufficiently large to comply with the condition in Lemma 1. Moreover,* τ_i *is the low pass filter time constant, which is to be selected small enough to render the behaviour of the input signal. In* (27b)*,* $r_{0,i}$ *imposes the convergence time. Therefore, it can be selected large to have fast convergence.* α_i *is sufficiently small and* ϵ_i *is sufficiently large such that* $|\dot{f}_{1,i}(\cdot)|/\alpha_i + \epsilon_i \geq a_{1,i}$*. Finally,* $q_i > 1$ *is the user-defined safety margin [84, 86].*

To analyze the stability and convergence, first the behaviour of equivalent control $\hat{w}_{i,eq}(t)$ in (27a) is studied in Lemma 4.

Lemma 4. *Consider the UAV translational dynamics* (20a) *and* (20b)*, represented as* (22)*, under the lumped disturbance vector* $\underline{f}_1(\cdot)$ *satisfying Assumption 1. Then the control law* (25)*, for sufficiently large* $k_{1,i}$*, with low-pass filter* (27a)*, guarantees* $\hat{w}_{i,eq}(t) = -\dot{f}_{1,i}(\cdot)$*, provided* $k_{2,i}(t) \geq a_{1,i}$*, for* $i \in \{x, y, z\}$*.*

Proof. By using the control law (25a) in the dynamics (22), it is easy to show that

$$\dot{\sigma}_i(t) = -k_{1,i}|\sigma_i(t)|^{0.5} sign(\sigma_i(t)) + \hat{w}_i(t) + f_{1,i}(\cdot), \tag{29}$$

for $i \in \{x, y, z\}$. Considering this with (25b) and taking into account Lemma 1, for $|\dot{f}_{1,i}(\cdot)| \leq a_{1,i}$ and $|\ddot{f}_{1,i}(\cdot)| \leq a_{2,i}$, provided $k_{2,i}(t) \geq a_{1,i}$, for sufficiently large $k_{1,i}$, both $\sigma_i(t)$ and $\hat{w}_i(t)$, converge to zero in finite time, i.e., it induces a second order sliding mode. On the other hand, (27a), for a small τ_i, is the low-pass filtered version of $\hat{w}_i(t)$, as explained in Remark 3. Therefore, considering Lemma 1, $\hat{w}_{i,eq}(t) = -\dot{f}_{1,i}(\cdot)$. □

As shown in Lemma 4, the equivalent control approaches to the time derivative of the disturbance, after the sliding surface is reached. Therefore, the time derivative of the equivalent control renders the second time derivative of disturbance. This justifies the inclusion of $a_{2,i}$ in (28b). Considering Lemma 4, for finite time convergence $\sigma_i(t)$ and $\hat{w}_i(t)$ to zero, it is required that $k_{2,i}(t) \geq a_{1,i}$. This is guaranteed by adaptive laws (27b) and (27c), proven in Lemma 5.

Lemma 5. *For $i \in \{x, y, z\}$, the adaptive laws* (27b) *and* (27c), *under condition $|d\hat{w}_{i,eq}(t)/dt| \leq q_i a_{2,i}$, force $\delta_i(t)$ and $e_i(t)$, defined in* (28), *to zero in finite time, which can be made arbitrarily small by the proper selection of the design parameter. Furthermore the gain $k_{2,i}(t)$ remains bounded. This further yields $k_{2,i}(t) \geq a_{1,i}$, for sufficiently small $0 < \alpha_i < 1$ and sufficiently large $\epsilon_i \in \mathbb{R}_+$.*

Proof. Using (27b), (27c) and (28), one can obtain that

$$\dot{\delta}_i(t) = -r_{0,i} sign(\delta_i(t)) + (e_i(t) - \frac{q_i}{\alpha_i} a_{2,i}) sign(\delta_i(t)) - \frac{1}{\alpha_i} \upsilon_i(t), \tag{30}$$

where $\upsilon_i(t) = |\frac{d\hat{w}_{i,eq}(t)}{dt}|$. Now, we define the a positive definite Lyapunov function as

$$V_i(t) = \frac{1}{2}\delta_i(t)^2 + \frac{1}{2\gamma_i} e_i(t)^2. \tag{31}$$

Therefore, using Lemma 3, it is readily shown that

$$\begin{aligned} \dot{V}_i(t) &\leq (e_i(t) - r_{0,i})|\delta_i(t)| - \frac{1}{\gamma_i} e_i(t)\dot{r}_i(t) \\ &= -\sqrt{2} r_{0,i} \left(\frac{1}{\sqrt{2}} |\delta_i(t)| + \frac{1}{\sqrt{2\gamma_i}} |e_i(t)| \right) \\ &\leq -\sqrt{2} r_{0,i} V_i(t). \end{aligned} \tag{32}$$

Then, considering Lemma 2, for any initial condition $V(0)$, $\delta_i(t)$ and $e_i(t)$ converge to zero in finite time smaller than $T\left(V(0)\right) = \sqrt{2V\left(0\right)}/r_{0,i}$. Furthermore, since $r_{0,i}$ is the design parameter, the convergence time can be made arbitrarily. Also note that since $e_i(t)$ is bounded, from (28b), $r_i(t)$ is bounded. This implies the boundedness of $k_{2,i}(t)$, considering (27c). Finite time convergence to zero implies $\delta_i(t) = 0$ for $t \geq T$. Therefore, considering (28a) and the result of Lemma 5, there exist sufficiently small $0 < \alpha_i < 1$ and sufficiently large $\epsilon_i \in \mathbb{R}_+$ such that $k_{2,i}(t) = \frac{1}{\alpha_i}|\hat{w}_{i,eq}(t)| + \epsilon_i = \frac{1}{\alpha_i}|\dot{f}_{1,i}(\cdot)| + \epsilon_i \geq a_{1,i}$. □

Now, the main properties of the proposed controller (25) is summarized in Theorem 1.

Theorem 1. *Consider the UAV translational dynamics* (20a) *and* (20b), *represented as* (22), *under the lumped disturbance vector $\underline{f}_1(\cdot)$ satisfying Assumption 1. Design the control law* (25), *for large $k_{1,i}$, $i \in \{x, y, z\}$, with the adaptive laws* (27), *for large constant $q_i > 1$, small $0 < \alpha_i < 1$ and large $\epsilon_i \in \mathbb{R}_+$. Then, for any initial condition $\underline{x}(0)$, the sliding surfaces* (24) *converges to zero in finite time, which can be adjusted by the user, and the UAV's position $\underline{x}(t)$ is stable and exponentially converges to the desired trajectory $\underline{x}_d(t)$.*

Proof. As proven in Lemma 5, $k_{2,i}(t) \geq a_{1,i}$ for $i \in \{x, y, z\}$, is satisfied. Then, considering Lemma 4, the finite time convergence of $\sigma_i(t)$ is guaranteed, which can be adjusted by proper selection of $r_{0,i}$. Consequently, considering (24), the exponential convergence of $\underline{x}(t)$ towards $\underline{x}_d(t)$ is obtained. □

Considering (28b), $a_{2,i}$ is required to be known. Even though this can be numerically obtained in the implementation, in some cases $a_{2,i}$ is unknown. This is studied in Theorem 2.

Theorem 2. *In the case of unknown $a_{2,i}$, the adaptive law* (27b) *is reformulated as*

$$\dot{r}_i(t) = \begin{cases} \gamma_i|\delta_i(t)| & |\delta_i(t)| \geq \delta_{0,i}, \\ 0 & otherwise, \end{cases} \tag{33}$$

for $i \in \{x, y, z\}$, where $\delta_{0,i}$ is a design parameter satisfying $\frac{\epsilon_i^2}{4} \geq \delta_{0,i}^2 + \frac{1}{\gamma_i}\left(\frac{q_i a_{2,i}}{\alpha_i}\right)^2$. Under Assumption 1, use the control law (25) *with the adaptive laws* (27a) *and* (27c). *Then the results of Theorem 1 are obtained.*

Proof. Use the Lyapunov function (31) in the analysis of $\delta_i(t)$ and $e_i(t)$. One can obtain that $\delta_i(t)\dot{\delta}_i(t) \leq (e_i(t) - r_{0,i})|\delta_i(t)|$. For $r_i(0) = 0$, we can see $r_i(t) \geq 0$. Then, for any pair $(\delta_i(t), e_i(t))$, $e_i(t) \leq \frac{q_i}{\alpha_i}a_{2,i}$ is satisfied, considering (28b). If $|\delta_i(t)| \geq \delta_{0,i}$, from (28), it is readily shown $\dot{e}_i(t) = -\gamma_i|\delta_i(t)|$ and, accordingly, $\dot{V}_i(t) \leq (e_i(t) - r_{0,i})|\delta_i(t)| - e_i(t)\dot{r}_i(t)/\gamma_i = -r_{0,i}|\delta_i(t)|$. On the other hand, for $|\delta_i(t)| \leq \delta_{0,i}$, if $e_i(t) \leq 0$, then $\dot{V}_i(t) \leq -r_{0,i}|\delta_i(t)|$. Let $E_i = \{(\delta_i(t), e_i(t)) : V_i(t) < \bar{r}_i\}$ be the smallest ellipsoid centred at the origin, with $\bar{r}_i = \delta_{0,i}^2/2 + (q_i a_{2,i}/\alpha_i)^2/2\gamma_i$, enclosing the rectangle $R_i = \{(\delta_i(t), e_i(t)) : |\delta_i(t)| \leq \delta_{0,i}, 0 \leq e_i(t) \leq q_i a_{2,i}/\alpha_i)\}$. It is readily shown that $R_i \subset E_i$. Since $\dot{r}_i(t) \geq 0$, then, $e_i(t) \leq q_i a_{2,i}/\alpha_i$ for all $t \geq 0$. Therefore, the solution outside of E_i, $\dot{V}_i(t) \leq -r_{0,i}|\delta_i(t)|$. Therefore, E_i is an invariant set. Let $\epsilon_i^2/4 \geq \delta_{0,i}^2 + (q_i a_{2,i}/\alpha_i)^2/\gamma_i$ be satisfied. If the pair $(\delta_i(t), e_i(t))$ enters E_i in finite time then it never leaves it and, in turn, $|\delta_i(t)| \leq \epsilon_i/2$. Otherwise, if it does not enter E_i, from $\dot{V}_i(t) \leq -r_{0,i}|\delta_i(t)|$, we obtain $\int_0^\infty r_{0,i}|\delta_i(t)|dt \leq V_i(0)$. Since $V_i(t)$ is bounded for $t \geq 0$, then solution $(\delta_i(t), e_i(t))$ is bounded. This implies $(\dot{\delta}_i(t), \dot{e}_i(t))$ is bounded, considering (33) and $\dot{e}_i(t) = -\dot{r}_i(t)$. Therefore, $\delta_i(t)$ and $|\delta_i(t)|$ are uniformly continuous. Taking Barbalat's Lemma [87], it is readily shown that $\lim_{t\to+\infty} \delta_i(t) = 0$ and there exists a finite time such that $|\delta_i(t)| \leq \epsilon_i/2$. Accordingly, whether or not whether If the pair $(\delta_i(t), e_i(t))$ enters E_i, $|\delta_i(t)| \leq \epsilon_i/2$ is guaranteed in finite time. Consequently, from (28a), it follows, $k_{2,i}(t) = \frac{1}{\alpha_i}|\hat{w}_{i,eq}(t)| + \epsilon_i/2 \geq a_{1,i}$ and a second order sliding mode is maintained. The rest of the proof is straightforward and therefore omitted. □

As stated in Theorem 1 and Remark 4, the gain $k_{1,i}$, $i \in \{x, y, z\}$, is constant and only required to be sufficiently large to maintain the sliding surface. However, as studied in [88], large $k_{1,i}$ might impose chattering. On the other hand, in [89] it has been suggested selecting the super twisting gains as $k_{1,i} = 1.5\sqrt{\bar{f}_{1,i}}$ and $k_{2,i} = 1.1\bar{f}_{1,i}$, where $|f_{1,i}| \leq \bar{f}_{1,i}$ to reduce the chattering. This convention has been commonly used in the literature. Taking this into account, therefore, here we design the gain $k_{1,i}$ as

$$k_{1,i}(t) = 1.5\sqrt{k_{2,i}(t)}, \tag{34}$$

for $i \in \{x, y, z\}$. This makes the gain $k_{1,i}$ large when the upper limit of the time derivative of disturbance, estimated by $k_{2,i}$, is increasing. Otherwise, it is reduced and an unnecessary large value for $k_{1,i}$ which might lead to chattering is avoided. It should be noted that, as proven in Lemma 5, the gain $k_{2,i}$ remains bounded. Thus, the boundedness of gain $k_{1,i}$, given by (34) is guaranteed.

Now, based on designed $u(t)$ in (22) and considering $u(t) = R(t)\underline{F}(t) = [u_x(t), u_y(t), u_z(t)]^T$ and (21), the thrust $f(t)$ and desired Euler angle $\underline{\Phi}_d(t) = [\theta_d(t), \phi_d(t), \psi_d(t)]^T$, for a given yaw angle $\psi_d(t)$, are obtained as

$$f(t) = \sqrt{u_x^2(t) + u_y^2(t) + u_z^2(t)}, \tag{35a}$$

$$\theta_d(t) = tan^{-1}\left(\frac{C_{\psi_d}(t)u_x(t) + S_{\psi_d}(t)u_y(t)}{u_z(t)}\right), \tag{35b}$$

$$\phi_d(t) = sin^{-1}\left(\frac{S_{\psi_d}(t)u_x(t) - C_{\psi_d}(t)u_y(t)}{f(t)}\right), \tag{35c}$$

respectively. Then, using (21b), the desired angular velocity $\underline{\omega}_d(t)$ is designed as

$$\underline{\omega}_d(t) = -\lambda_\omega R_q(\underline{\Phi}(t))\left(\underline{\Phi}(t) - \underline{\Phi}_d(t)\right), \tag{36}$$

where $\lambda_\omega \in \mathbb{R}_+$ is a design parameter. The control design procedure is given in Algorithm 2.

Remark 5. *The onboard controller computes desired angular rates using the error between desired and actual Euler angles at the frequency of 250 Hz, using proportional controller. The onboard angular rate controller is a PID controller that runs at 1000 Hz to compute scaled $\underline{\tau}(t)$ in the form of the motor PWM signals*[1]*. The tracking performance and robustness of the onboard angular rate controller have been already proven [90]. Additionally, desired angular rate is also computed offboard using* (36) *at the frequency of 50 Hz and it is feed-forwarded to the onboard angular rate controller to improve tracking performance.*

Algorithm 2 ASTC Design Algorithm

```
1:  Inputs: $\underline{x}(t)$, $\underline{v}(t)$, $\underline{x}_d(t)$
2:  for $i \in \{x, y, z\}$ do
3:      Select $\lambda_i$, $\tau_i$, $\gamma_i$, $r_{0,i}$, $\alpha_i$, $\epsilon_i$ and $q_i$, as per Remark 4,
4:      Compute sliding surface $\sigma_i(t)$, as (24),
5:      if $a_{2,i}$ is known then
6:          Take integral of (27) and (25b), using (28),
7:      else
8:          Take integral of (27a), (27b), (33) and (25b), using (28),
9:      end if
10:     Compute $k_{1,i}(t)$ as (34),
11:     Compute $u_i(t)$ as (25a),
12: end for
13: Compute $f(t)$ and $\underline{\Phi}_d(t)$, as (26), for given $\psi_d(t)$,
14: Compute $\underline{\omega}_d(t)$ as (36),
15: Compute PWM using onboard angular rate control.
```

[1]https://bit.ly/3SshggW

4 Perception-Aware Trajectory Planning and Tracking: Validation

In this section, the high-fidelity numerical simulations in Gazebo as well as experimental results are presented, to evaluate the performance of the proposed algorithms. This is to validate the proposed approaches as a proof of concept. To execute the proposed examples, both simulations and experiments, Algorithms 3-7 are given in Appendix A.

4.1 Validation methodology and setup

To validate the proposed solutions, we conduct Gazebo simulations and in-lab experiments. The overall diagram of the architecture of the validation scenario is shown in Fig 14.

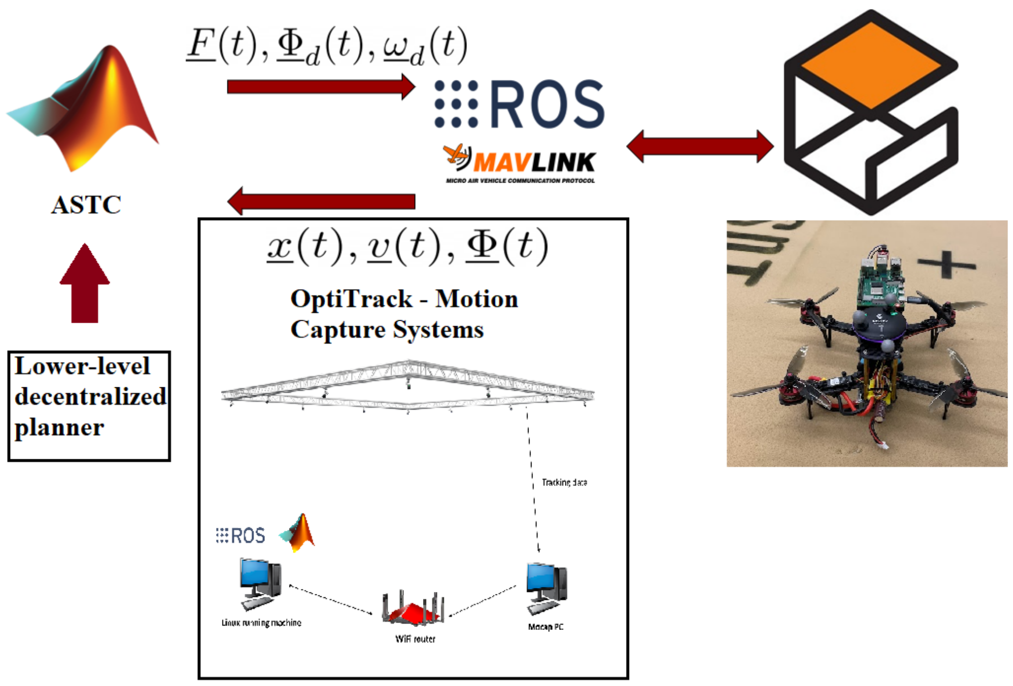

$\underline{F}(t), \underline{\Phi}_d(t), \underline{\omega}_d(t)$
ROS
MAVLINK
MICRO AIR VEHICLE COMMUNICATION PROTOCOL
ASTC
$\underline{x}(t), \underline{v}(t), \underline{\Phi}(t)$
OptiTrack - Motion Capture Systems
Lower-level decentralized planner
Tracking data
ROS
Linux running machine
WiFi router
Mocap PC

Figure 14: The overall architecture of the validation scenario.

It is worth noting that in ASTC the computed $f(t)$ is scaled to $[0, 1]$, representing the UAV thrust range since the low-level control takes the normalized thrust to compute the PWM. The scaling factor is obtained from the Gazebo simulation by hovering UAV at a fixed point and monitoring the MAVROS topic `/mavros/setpoint_raw/target_local`, as explained in Algorithm 5. The control parameters are selected as $\lambda_i = 2$, $\tau_i = 0.02$, $\gamma_i = 2$, $r_{0,i} = 0.0001$, $\alpha_i = 0.9$, $\epsilon_x = 0.005$, $\epsilon_y = 0.005$, $\epsilon_z = 0.01$, $q_x a_{2,x} = 0.04$, $q_y a_{2,y} = 0.04$ and $q_z a_{2,z} = 0.01$ for $i \in \{x, y, z\}$. The considered desired trajectories are the circle and lemniscate, defined as $\underline{x}_d(t) = [Rcos(2\pi t/T), Rsin(2\pi t/T), 1]^T$ and $\underline{x}_d(t) = [Rcos(2\pi t/T), Rsin(\pi t/T)cos(2\pi t/T), 1]^T$, respectively, with radius R and time period T of trajectory. In the numerical simulations, the disturbance in (20b), is modelled as

$$\underline{f}_d(\cdot) = \begin{bmatrix} -\frac{C_p \rho v_{x,w}(t)}{2\sqrt{v_{x,w}(t)^2 + v_{y,w}(t)^2}} \\ -\frac{C_p \rho v_{y,w}(t)}{2\sqrt{v_{x,w}(t)^2 + v_{y,w}(t)^2}} \\ 0 \end{bmatrix},$$

where $C_p = 0.1$ is drag coefficient, and $\rho = 1.225(kg/m^3)$ is the air density. Also, $v_{x,w}(t) = sin(0.75\pi t/T)$ and $v_{y,w}(t) = cos(0.75\pi t/T)$ are the wind speed in x and y directions, respectively. $\underline{f}_d(\cdot)$ represents the drag effects which depend on the wind speed [91]. To evaluate the performance of the proposed ASTC for high-speed maneuvers of UAV, the time period T of the desired trajectory is time variable over 90 seconds of simulation time, defined as

$$T = \begin{cases} 8\pi, & 0 < t < 15 \\ 8\pi - \frac{5\pi(t-15)}{30}, & 15 < t < 45 \\ 3\pi + \frac{5\pi(t-45)}{30}, & 45 < t < 75 \\ 8\pi. & 75 < t < 90 \end{cases}$$

Moreover, the radius R is selected as $4m$. The shortest time period is selected to be 3π, which corresponds to the maximum speed, reachable by the considered UAV. The desired yaw angle is defined as $\psi_d(t) = arctan(v_y(t)/v_x(t))$. For the sake of brevity, and to compare the numerous results, the Root Mean Square (RMS) error is considered, defined as $e_{RMS}(t) = \|\underline{x}(t) - \underline{x}_d(t)\|$.

On the other hand, to evaluate the performance of the proposed trajectory planning algorithm, in the Gazebo simulations, we have considered imaginary stationery and moving obstacle to be avoided by the drone. Moreover, the initial and final points are provided. In the experimental setup, we have considered the same scenario for large and small drones. Finally, in MRS experimental setup, we used a small drone, to track a circular trajectory using ASTC. Then, this drone is considered as an obstacle for the other drone to be avoided. Here we have taken for granted that the position of the obstacle, the target drone, the initial and final points are given. In fact, this information is provided by the Collaborative Perception and Sensor Fusion component. Therefore, it is logical to assume this information is provided. In the experiments, this information is provided by the motion capture system. Finally, in the MRS trajectory planning optimization, we always keep the field of view of the target drone towards the obstacle, to be able to capture the perceptive information as much as possible, which was tackled in Task 2.3.

4.2 Gazebo simulation results and discussion

The results of ASTC are shown in Figures 15-24. To highlight the superiority of the designed ASTC with both adaptive $k_{1,i}$ and $k_{2,i}$, for $i \in \{x, y, z\}$, the results of STC and traditional ASTC are also studied. For STC, the gains are selected as $k_{1,i} = 1.5\sqrt{\Delta_i}$ and $k_{2,i} = 1.1\Delta_i$, [89], with $\Delta_x = \Delta_y = 0.04$ and $\Delta_z = 0.002$. It should be noted that these design parameters are obtained via trial and error to achieve the best tracking performance. Also for traditional ASTC the same $k_{1,i}$ is used while only $k_{2,i}$ is adaptive.

Considering Figures 15, 16, 20 and 21, the proposed controller is able to steer the UAV position towards the desired trajectories with variable speed. The desired trajectory reaches to its maximum speed at 45 seconds. More importantly, Figures 17 and 22 confirm that $k_{2,i}(t) \geq a_{1,i}$, for $i \in \{x, y, z\}$, as proven in Lemma 5. This leads to disturbance rejection. It should be noted that there is an inherent unknown drag model and uncertainties, e.g. measurement uncertainty, model mismatch and communication lag, imposed in the Gazebo simulations. This causes larger value for $k_{2,i}(t)$, compared to the time derivative of applied disturbance $\underline{f}_d(\cdot)$. This is more obvious considering the variation of $k_{2,z}(t)$ in Figures 17 and 22, while no exogenous disturbance is applied in z direction. Compared to the STC and traditional ASTC, given in Figures 18 and 23, the fast convergence of the proposed controller is seen in the presence of disturbance. This represents one main characteristic of the proposed controller. Even though the similar tracking performance is achieved by using the STC and traditional ASTC at high speed maneuvers, the chattering is significant for these two controllers for low speed, considering the Euler angles, shown in Figures 19 and 24. The chattering issue is avoided using the proposed controller for both low and high speed situations. This further highlights the advantage of the proposed controller.

Now, the Gazebo simulations are shown for a single drone trajectory planning is shown in Figures 25 and 26 . It is obvious that the optimized trajectory is always kept outside of the safety region around the obstacle. More

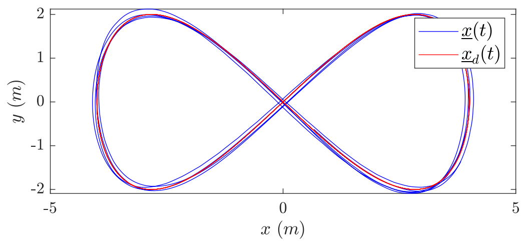


Figure 15: UAV position for lemniscate trajectory using the proposed controller.

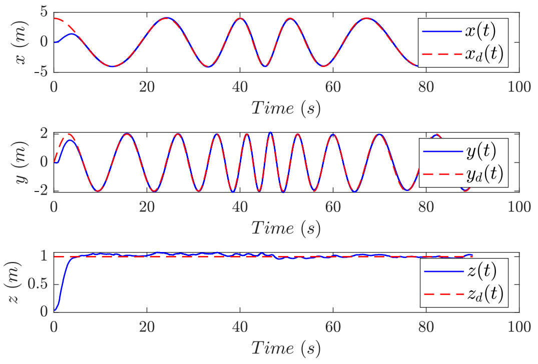


Figure 16: Lemniscate tracking performance using the proposed controller.

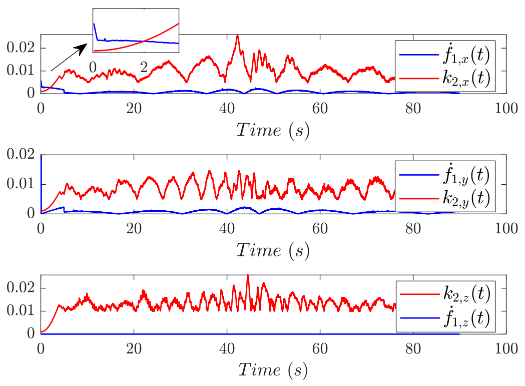


Figure 17: Adaptive gains using the proposed controller for lemniscate trajectory.

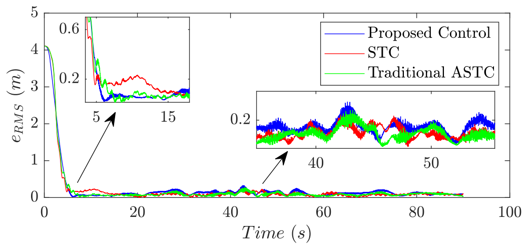


Figure 18: RMS comparison for lemniscate trajectory.

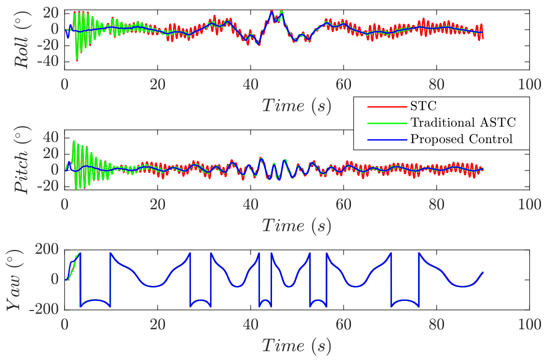


Figure 19: Euler angles comparison for lemniscate trajectory.

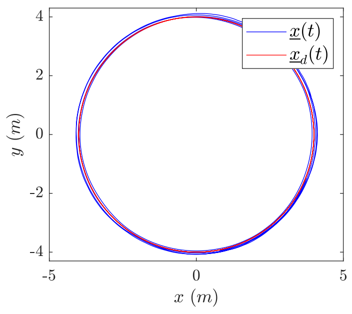


Figure 20: UAV position for circle trajectory using the proposed controller.

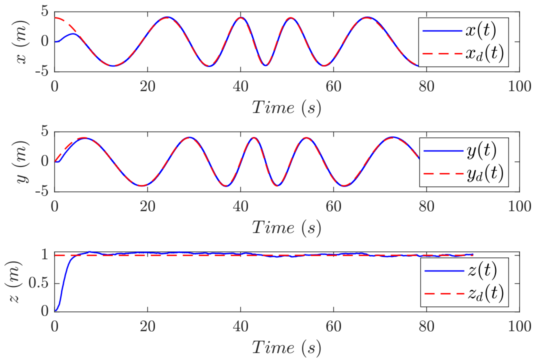


Figure 21: Circle tracking performance using the proposed controller.

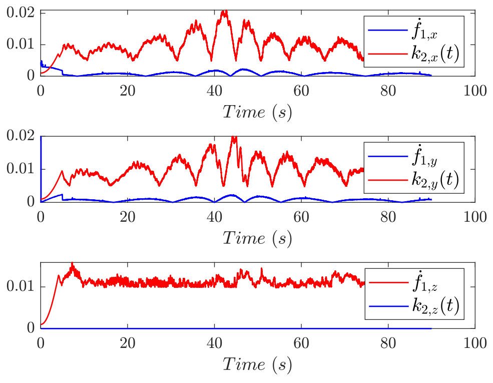


Figure 22: Adaptive gains using the proposed controller for circle trajectory.

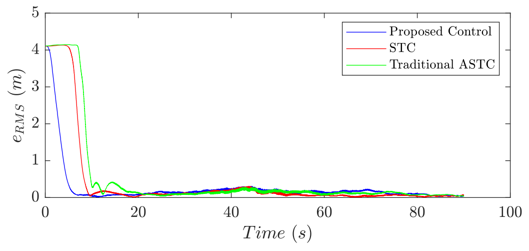


Figure 23: RMS comparison for circle trajectory.

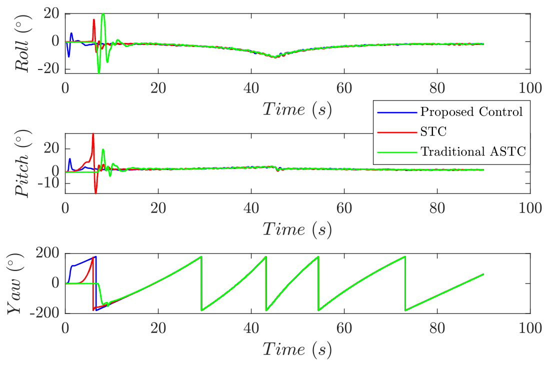


Figure 24: Euler angles comparison for circle trajectory.

importantly, the effect of the repulsive force from the obstacle is illustrated in Figure 26. Also, the obstacle is always kept in the field of view of the drone.

4.3 Experimental results and discussion

The experimental results of ASTC are shown in Figures 27-32. For the experimental setup, the desired trajectories include circle and lemniscate paths with the same time period and radius as we used in the Gazebo simulation. Moreover, different external wind disturbances are applied using a fan with three different speeds, namely, no disturbance (S0), slow (S1), medium (S2) and high (S3) speeds. For the sake of brevity the mean, standard deviation (std) and maximum (max) of RMS tracking error for different disturbance speeds and trajectories are summarized in Tabel 2. Considering 27-32, the proposed control is able to track accurately different trajectories with variable speed. Moreover, after the sliding surface is reached, the tracking error is kept within the region around zero, for different disturbances. This is further confirmed considering Table 2, in which the mean, std and maximum of RMS for different trajectories and different wind disturbances are summarized. It is obvious that the metrics are kept the same in different situations, owning to the adaptive gains of the proposed control.

Table 2: Experimental RMS results in centimeter for different disturbance and trajectories.

	Circle Trajectory			Lemniscate Trajectory		
RMS	mean	std	max	mean	std	max
S0	0.04	1.02	6.92	0.21	0.85	8.90
S1	0.30	1.06	5.80	0.23	0.90	8.57
S2	0.25	0.92	5.44	0.23	0.92	8.77
S3	0.23	0.84	5.04	0.24	0.91	8.58

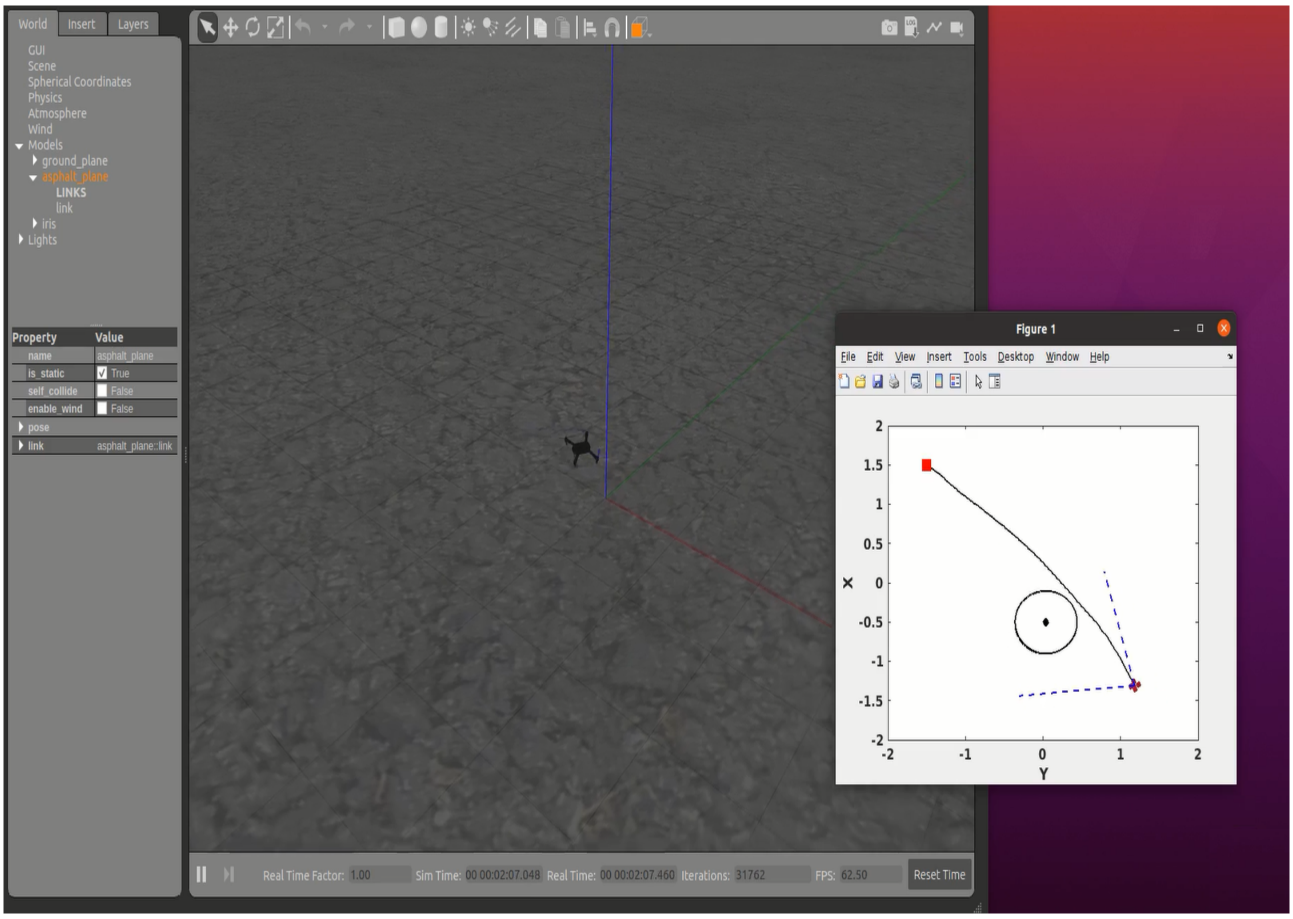


Figure 25: Gazebo simulation results for single drone. The red square is the final point to be reached by drone. The black dot denotes the position of the obstacle. The black circle represents the safety region around the obstacle. The dashed blue lines show the field of view of the drone.

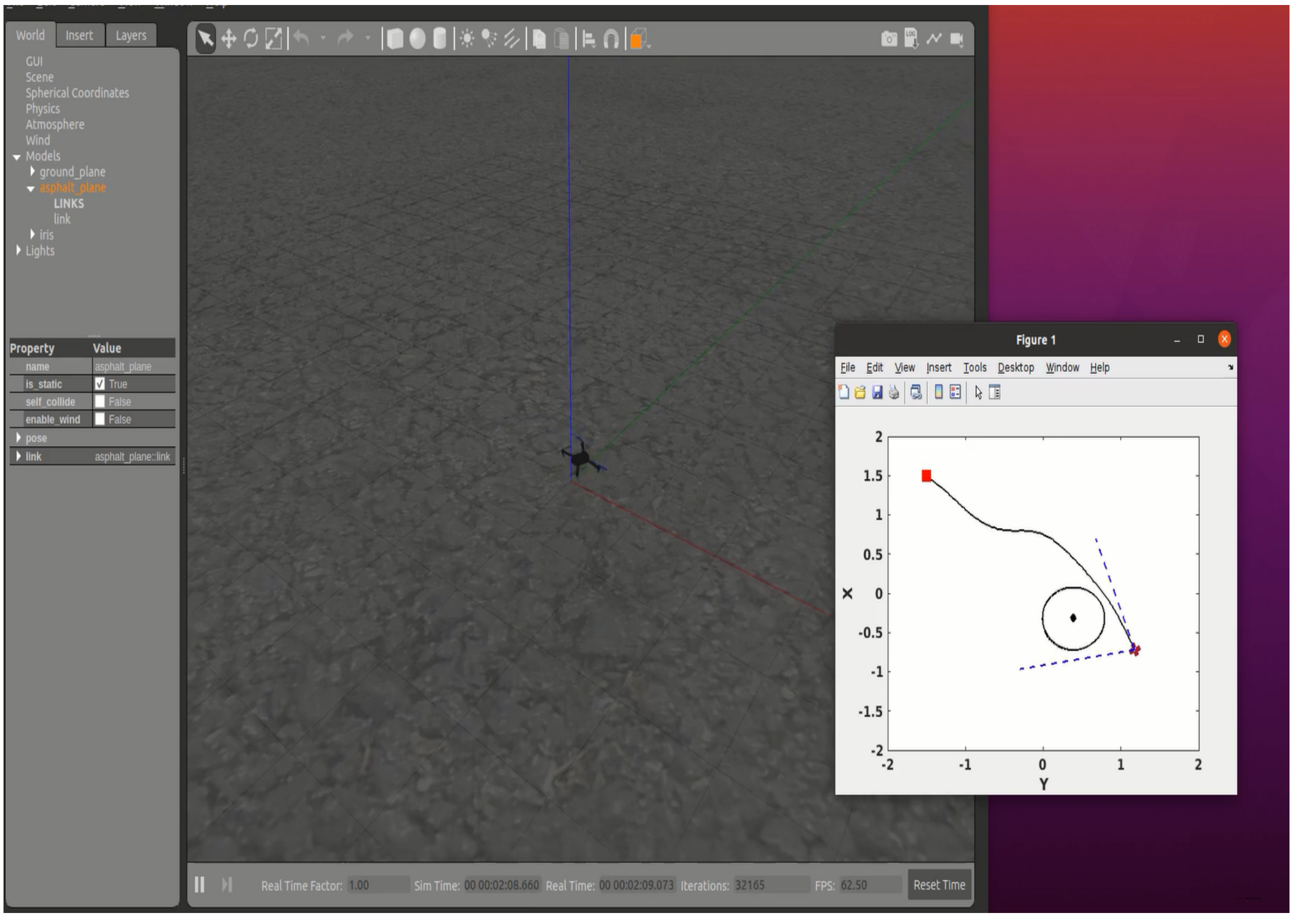


Figure 26: Effect of repulsive force on trajectory optimization in Gazebo simulation for single drone. The black dot denotes the position of the obstacle. The black circle represents the safety region around the obstacle. The dashed blue lines show the field of view of the drone.

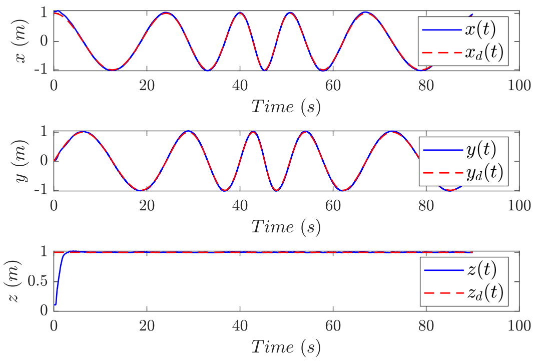


Figure 27: Experimental tracking performance for circle trajectory with S3 disturbance.

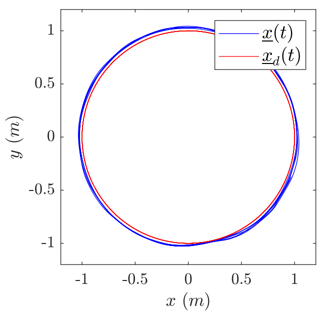


Figure 28: Experimental UAV position for circle trajectory with S3 disturbance.

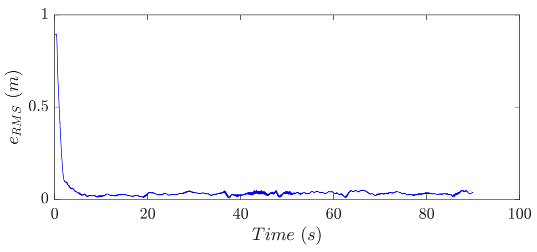


Figure 29: Experimental RMS for circle trajectory with S3 disturbance.

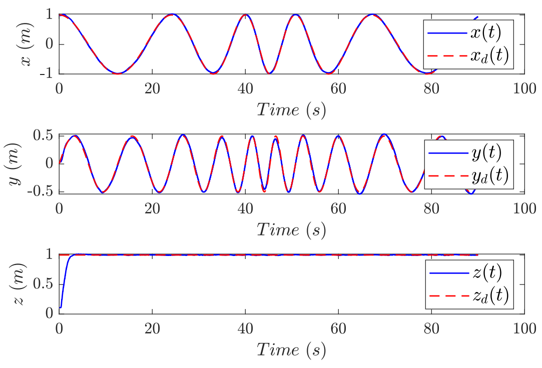


Figure 30: Experimental tracking performance for lemniscate trajectory with S3 disturbance.

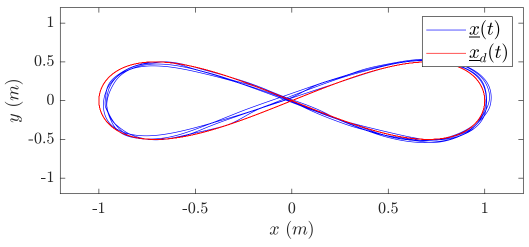


Figure 31: Experimental UAV position for lemniscate trajectory with S3 disturbance.

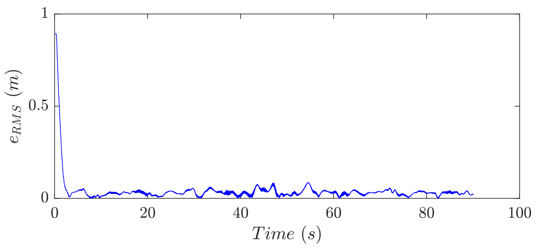


Figure 32: Experimental RMS for lemniscate trajectory with S3 disturbance.

In Figure 33 the experimental result of a single drone trajectory planning is shown. The re-optimized trajectories are shown to illustrate how the trajectory is being updated during the maneuver. Even though the obstacle is stationary the due to inherent uncertainty in the measurements of the drone and obstacle positions, at each re-optimization step the obtained trajectory is slightly changed. This represents the significant feature of the proposed algorithm that takes the updated measurements into account, which enables the overall solution to deal with the uncertainty.

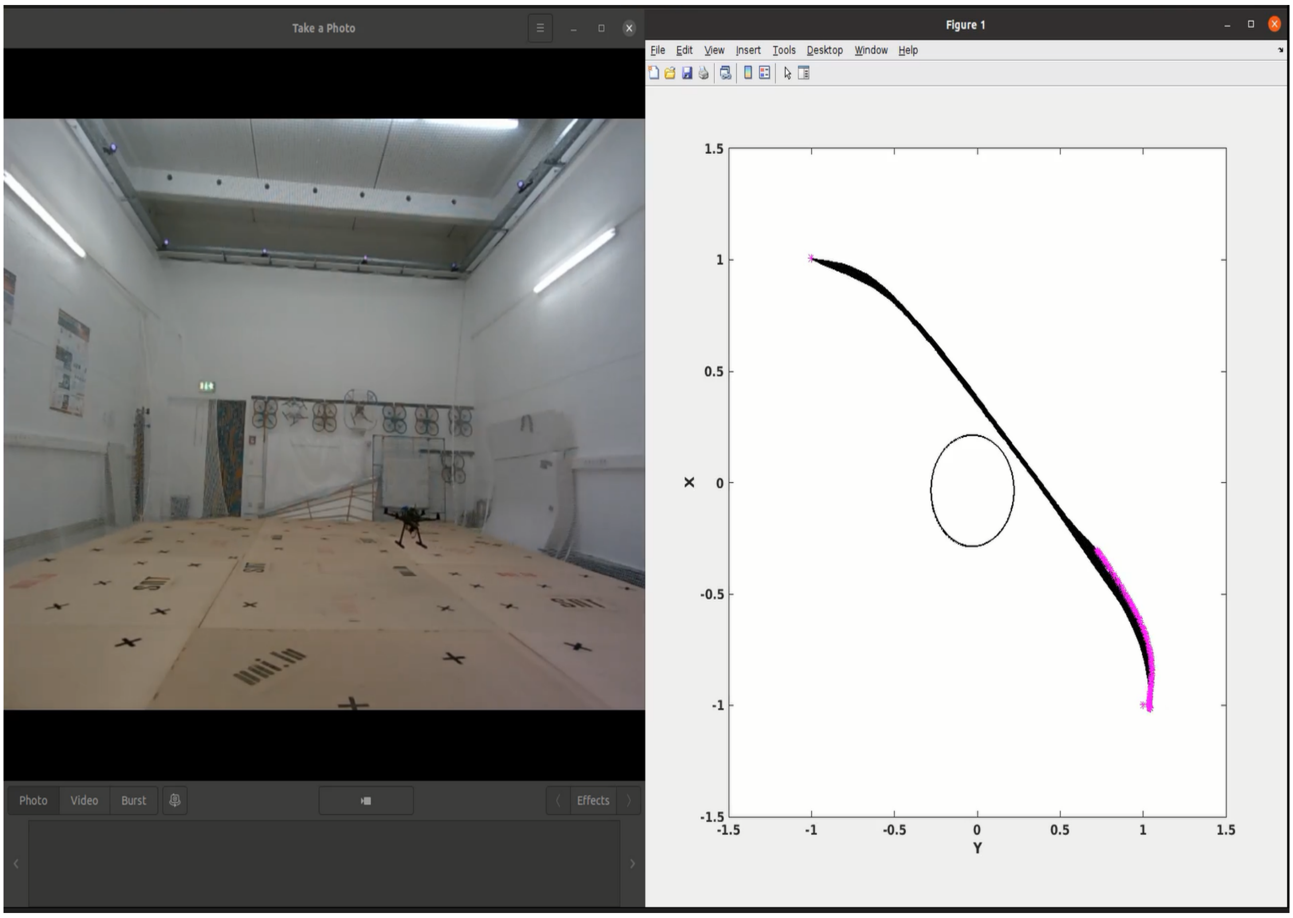


Figure 33: Experimental result of trajectory optimization for a large drone with stationary obstacle and no perception constraints. The black circle represents the safety region around the obstacle. The black lines represent the optimized and real-time re-optimized trajectories. The purple points shown the drone position tracking the trajectory.

In Figures 34-36 the trajectory planning experimental results are shown for MRS. We used a small drone, to track a circular trajectory using ASTC. Then, this drone is considered as an obstacle for the other drone to be avoided. Here we have taken for granted that the position of the obstacle, the target drone, the initial and final points are given. In fact, this information is provided by the Collaborative Perception and Sensor Fusion component. In the MRS trajectory planning optimization, we always keep the field of view of the target drone towards the obstacle, to be able to capture the perceptive information as much as possible, which was tackled in Task 2.3. It is obvious that the optimized trajectory is always kept outside of the safety region around the obstacle. Also, the obstacle is always kept in the field of view of the drone. Due to limited space, we interchange the initial and final points as soon as the drone reaches there. However, in practice, the waypoints can be set all at once or be sent to the drone sequentially.

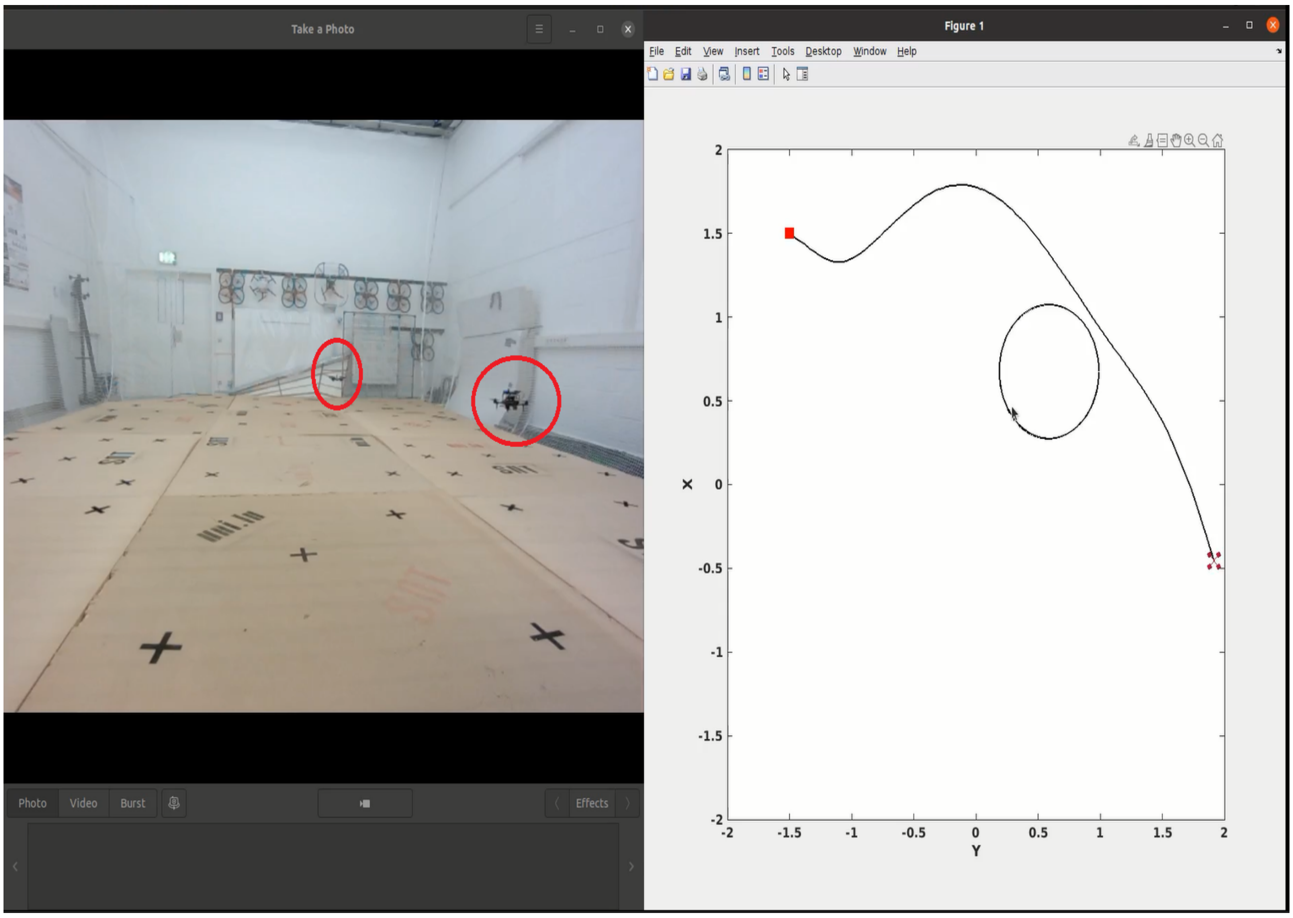


Figure 34: Experimental result of MRS trajectory optimization with moving obstacle and no perception constraints. The black circle represents the safety region around the obstacle. The black lines represent the optimized and real-time re-optimized trajectories.

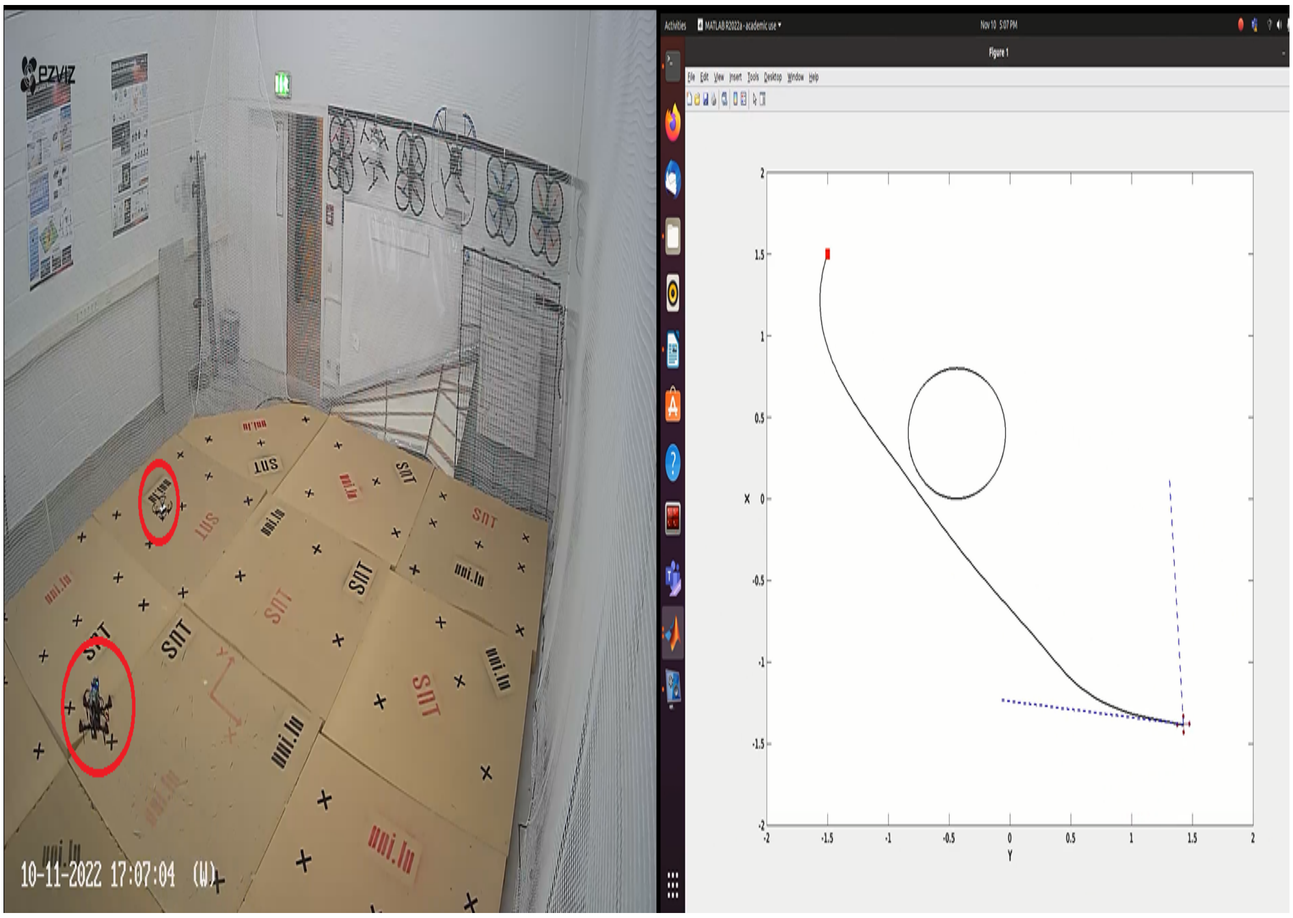


Figure 35: Experimental result of MRS trajectory optimization with moving obstacle and with perception constraints, from initial for final points. The black circle represents the safety region around the obstacle. The black lines represent the optimized and real-time re-optimized trajectories. The dashed blue lines show the field of view of the drone.

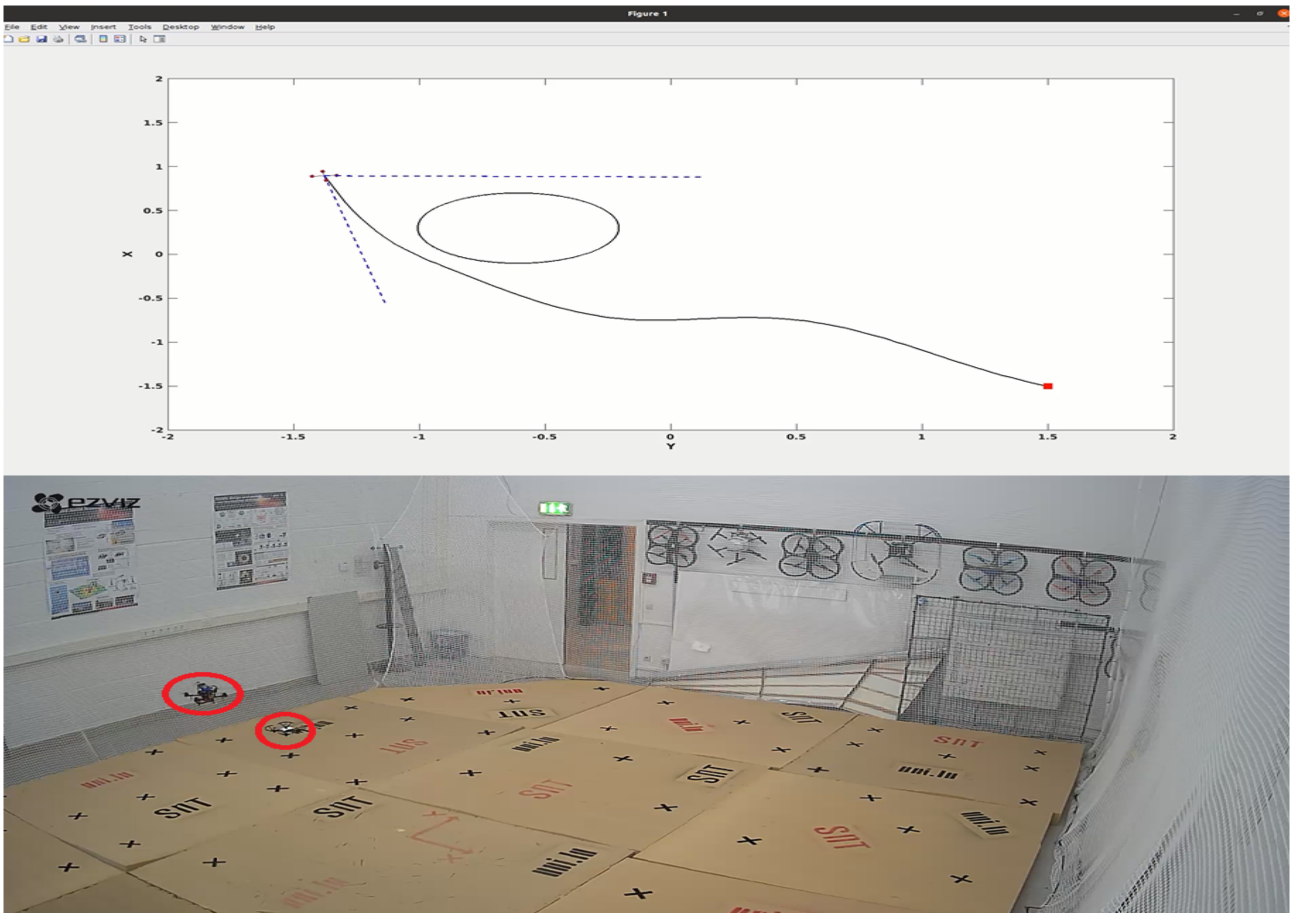


Figure 36: Experimental result of MRS trajectory optimization with moving obstacle and with perception constraints from final to initial points. The black circle represents the safety region around the obstacle. The black lines represent the optimized and real-time re-optimized trajectories. The dashed blue lines show the field of view of the drone.

5 Future Works

In the next steps of this research, aligning with the SESAME project objectives, we conduct the following steps to accomplish the aims for autonomous safe navigation of MRS for targeting M30.

- Integration of ASTC with trajectory optimization to have both planning and tracking components be executed at the same time.
- The novel approach to generate trajectories of a collaborative MRS performed in Task 2.4. The generated trajectories can provide extra off-board sensor feedback to rescue robots under various distress types (e.g., cyber-attack, sensor malfunction).
- Computing the optimal sensitivity with respect to parameters (initial position and obstacle position) via sIPOPT to have faster and computationally less expensive algorithms to run on onboard computers.
- Generic obstacle representation (ellipsoid, squares)
- Different problems and robot types to be considered

6 Potential applicability for use cases

In this section, the potential applicability of the proposed approaches for different use cases is briefly motivated.

- Use Case 1: Dependable Multi-Robot Systems in the Battery Innovation Centre: For this use case, the battery parts are to be picked and assembled, can be geometrically considered as the waypoints, through which the manipulator end effector is to be navigated. Also, the other manipulator or human operator can be considered as obstacles to be avoided. Therefore, the trajectory planner can be utilized here. Moreover, the ASTC can be individually used for the control of the manipulators, even if the trajectory is already preassigned.
- Use Case 2: Disinfecting Hospital Environments using Robotic Teams: The ground mobile robot can take advantage of the proposed planners, given the implemented constraints in the algorithm. The constraints might represent the area the robot is to visit or avoid. More importantly, the presented results can be directly obtained for the ground robot, as we have fixed the altitude of the drone which can represent a planar motion similar to the ground robot. Also, the fastest trajectory can lead to fast disinfection to avoid intervention with other personnel in the hospital environment.
- Use Case 3: Power Station Inspection using Autonomous Multi-Robot Systems: Here, the obtained results can be directly used for this use case, as we have evaluated the proposed algorithm on drones. Furthermore, the proposed ASTC is a good fit, given the accuracy of the navigation and the inherent wind disturbance or uncertainties in the site. Also, the initial and final inspection points can be fed into the planning approach to find a safe trajectory to be followed for autonomous inspection.
- Use Case 4: Autonomous Pest Management in Viticulture: Here, the monitoring drone is to fly over the farm to detect the plants which need to be sprayed or the locations of obstacles. Therefore, for this task, the given path is to cover the whole farm, in which ASTC can be used individually. Consequently, the locations of plants to be sprayed and obstacles are fed sequentially or all at once into the trajectory planner of the spraying drone, for which a trajectory optimization problem is solved in real-time. Then, the spraying drone is to follow this trajectory, for which ATSC can be used separately.

7 Conclusions

This document outlined the components for Perception-Aware Trajectory Planning and Tracking for MRS. For the planning part, we developed an online trajectory optimization approach to compute the fastest trajectory, given the initial and final positions. We used Ipopt for solving the problem. We proposed some improvements in the modelling of the optimization problem to speed up the solver. Moreover, we considered the obstacle avoidance issue as a safety region around the obstacle. We kept the position and field of view to the given area, satisfying the maximization of capturing the perceptive information. The tracking control design was tackled in the presence of external disturbance for agile maneuver using adaptive super-twisting control. The proposed controller provides the finite time convergence and attenuation of the disturbance. Moreover, the assumption on the variation of disturbance is relaxed by designing an adaptive law. Furthermore, we provided the algorithms and instructions to execute the codes. The efficiency of the proposed approaches was investigated by high-fidelity simulations in Gazebo as well as experimental studies.

A Algorithms and Instructions for Codes Execution

In this appendix, the instructions to execute the codes for the examples are given. It should be noted that these algorithms are just to reproduce the results of the given simulations and experiments in Section 4, not the use case integration, which is addressed in WP8. To be able to run the provided code packages the following software are to be installed.

- Ubuntu 20.04.3 LTS `http://http://lt.releases.ubuntu.com/20.04.3/`
- MATLAB R2022a `https://nl.mathworks.com/help/install/ug/get-new-matlab-release.html`
- Ipopt `https://coin-or.github.io/Ipopt/INSTALL.html`
- ROS Noetic `http://wiki.ros.org/noetic/Installation/Ubuntu`
- PX4 `https://docs.px4.io/main/en/dev_setup/dev_env_linux_ubuntu.html`
- CasADi `https://web.casadi.org/get/`

The executions of numerical simulations and experimental codes for both ASTC and trajectory optimization are elaborated in Algorithms 3-7. The instructions given in these algorithms are to be followed exactly as presented.

Algorithm 3 Gazebo Simulation Execution of ASTC Codes

```
 1: Open a new terminal
 2: Run
    cd Firmware          ▷ Change the directory to the folder where PX4 is installed.
 3: Run
    roslaunch mavros px4.launch fcu_url:="udp://:14540@127.0.0.1:14540"
 4: Open a new tab in the same terminal and run
    make px4_sitl gazebo
 5: Open a new tab in the same terminal and run
    rostopic pub -r 100 /traj std_msgs/Float32MultiArray "layout:
    dim:
    - label: ''
    size: 0
    stride: 0
    data_offset: 0
    data: [1, 6.28]"
 6: Open a new tab in the same terminal and run
    rostopic pub -r 100 /smc_gains std_msgs/Float32MultiArray "layout:
    dim:
    - label: ''
    size: 0
    stride: 0
    data_offset: 0
    data: [1.0, 0.03, 1, 0.002, 0.1, 0.1, 14]"
 7: Open a new tab in the same terminal and run
    rostopic pub -r 100 /pos_des std_msgs/Float32MultiArray "layout:
    dim:
    - label: ''
    size: 0
    stride: 0
    data_offset: 0
    data: [0.0, 0.0, 3.0, 0.0]" ▷ For fixed position control, different desired positions
    can be set in the format of data: [$x_d$, $y_d$, $z_d$, $\psi_d$]|.
 8: Open MATLAB and change the address bar to the thrust_attitude_control_Ver_5
 9: In MATLAB command window run
    rosinit
10: Open the file main_th_att_control_astc_2gains2.m
11: Uncomment the lines between 178-186 (fixed point), 213-222 (circular trajectory) or 237-246
    (lemniscate trajectory)
12: Run main_th_att_control_astc_2gains2.m file.
```

Algorithm 4 Gazebo Simulation Execution of Trajectory Optimization Codes for Single Drone

```
1: Open a new terminal
2: Run
   cd Firmware                    ▷ Change the directory to the folder where PX4 is installed.
3: Run
   roslaunch mavros px4.launch fcu_url:="udp://:14540@127.0.0.1:14540"
4: Open a new tab in the same terminal and run
   make px4_sitl gazebo
5: Open MATLAB and change the address bar to the to_blf_perception
6: Open the file main.m
7: In MATLAB command window run
   rosinit
8: Run main.m file.
```

Algorithm 5 Experimental ASTC Codes

```
 1: Open a new terminal
 2: Run
    Ifconfig        ▷ This is to check the IP address of the computer and the onboard computers.
 3: Run
    ssh pi@192.168.30.146        ▷ Replace 192.168.30.146 with the onboard computer IP.
 4: Run
    export ROS_MASTER_URI=http://192.168.30.146:11311 ▷ Replace 192.168.30.146 with
    the onboard computer IP.
 5: Run
    export ROS_IP=192.168.30.146        ▷ Replace 192.168.30.146 with the onboard computer IP.
 6: Run
    screen
 7: Run
    roscore
 8: Press
    Ctrl+A+C
 9: Run
    roslaunch mavros px4.launch fcu_url:=/dev/ttyUSB0:921600 gcs_url:=udp://@192.168.30.125:14550
    ▷ Replace 192.168.30.125 with the computer IP.
10: Press
    Ctrl+A+C
11: Run
    roslaunch vrpn_client_ros sample.launch server:=192.168.30.105
    ▷ Replace 192.168.30.105 with IP of Optitrack or the other UAV providing the pose estimation of main
    UAV.
12: Press
    Ctrl+A+C
13: Run
    rosrun topic_tools relay /vrpn_client_node/quail/pose /mavros/vision_pose/pose
    ▷ Replace quail with the body name given in the Optitrack.
14: Press
    Ctrl+A+C
15: Run
    rostopic echo /mavros/local_position/pose
    ▷ This is to check if the position of the UAV is displayed. If so, press Ctrl+C to terminate this.
16: Open a new terminal
17: Run
    export ROS_MASTER_URI=http://192.168.30.146:11311
    ▷ Replace 192.168.30.146 with the onboard computer IP.
18: Run
    export ROS_IP=192.168.30.125
    ▷ Replace 192.168.30.125 with the computer IP.
19: Open MATLAB and change the address bar to the astc_experiments
20: In MATLAB command window run
    rosinit
21: Open the file main_th_att_control_astc.m
22: Uncomment the lines between 178-188 (fixed point) and comment out the other trajectories, i.e., the lines
    between 191-200 and 203-212 must be commented.
```

```
23: Open a new tab in the terminal and run
    rostopic pub -r 100 /traj std_msgs/Float32MultiArray "layout:
    dim:
    - label: ''
    size: 0
    stride: 0
    data_offset: 0
    data: [1, 6.28]"
24: Open a new tab in the terminal and run
    rostopic pub -r 100 /smc_gains std_msgs/Float32MultiArray "layout:
    dim:
    - label: ''
    size: 0
    stride: 0
    data_offset: 0
    data: [1.0, 0.03, 1, 0.002, 0.1, 0.1, 14]"
    ▷ The number 14 in the last line will be amended later on considering the model of UAV is used.
25: Open a new tab in the terminal and run
    rostopic pub -r 100 /pos_des std_msgs/Float32MultiArray "layout:
    dim:
    - label: ''
    size: 0
    stride: 0
    data_offset: 0
    data: [0.0, 0.0, 3.0, 0.0]"    ▷ For fixed position control, different desired positions can be
    set in the format of data: [$x_d$, $y_d$, $z_d$, $\psi_d$]|.
26: Run main_th_att_control_astc.m file.
27: Open a new tab in the terminal and run
    rostopic echo /mavros/target_actuator_control
28: In the results shown in the terminal, in the control line, check the $4^{th}$ number. Divide 9.81 by that number.
    Replace that result instead of the number 14 above (on line 24).
29: Uncomment the other trajectories, i.e., the lines between 191-200 or 203-21 and run
    main_th_att_control_astc.m file.
```

Algorithm 6 Experimental Trajectory Optimization Codes for Single UAV

```
1:  Open a new terminal
2:  Run
    Ifconfig          ▷ This is to check the IP address of the computer and the onboard computers.
3:  Run
    ssh pi@192.168.30.146          ▷ Replace 192.168.30.146 with the onboard computer IP.
4:  Run
    export ROS_MASTER_URI=http://192.168.30.146:11311          ▷ Replace
    192.168.30.146 with the onboard computer IP.
5:  Run
    export ROS_IP=192.168.30.146          ▷ Replace 192.168.30.146 with the onboard
    computer IP.
6:  Run
    screen
7:  Run
    roscore
8:  Press
    Ctrl+A+C
9:  Run
    roslaunch mavros px4.launch fcu_url:=/dev/ttyUSB0:921600 gcs_url:=udp://@192.168.30.125:14550
    ▷ Replace 192.168.30.125 with the computer IP.
10: Press
    Ctrl+A+C
11: Run
    roslaunch vrpn_client_ros sample.launch server:=192.168.30.105
    ▷ Replace 192.168.30.105 with IP of Optitrack or the other UAV providing the pose estimation
    of main UAV.
12: Press
    Ctrl+A+C
13: Run
    rosrun topic_tools relay /vrpn_client_node/quail/pose /mavros/vision_pose/pose
    ▷ Replace quail with the body name given in the Optitrack.
14: Press
    Ctrl+A+C
15: Run
    rostopic echo /mavros/local_position/pose
    ▷ This is to check if the position of the UAV is displayed. If so, press Ctrl+C to terminate this.
16: Open a new terminal
17: Run
    export ROS_MASTER_URI=http://192.168.30.146:11311
    ▷ Replace 192.168.30.146 with the onboard computer IP.
18: Run
    export ROS_IP=192.168.30.125
    ▷ Replace 192.168.30.125 with the computer IP.
19: Open MATLAB and change the address bar to the to_blf_perception
20: In MATLAB command window run
    rosinit
21: Open and run the file main.m
    ▷ In lines 63 and 64, you can change the positions of initial and final points. Also, in line 66 the
    last number, you can change the safety radius around the obstacle.
```

Algorithm 7 Experimental Trajectory Optimization Codes for MRS

```
 1: Open a new terminal
 2: Run
    Ifconfig    ▷ This is to check the IP address of the computer and the onboard computers.
 3: Run
    ssh pi@192.168.30.146    ▷ Replace 192.168.30.146 with the onboard computer IP.
 4: Run
    export ROS_MASTER_URI=http://192.168.30.146:11311 ▷ Replace 192.168.30.146 with
    the onboard computer IP.
 5: Run
    export ROS_IP=192.168.30.146    ▷ Replace 192.168.30.146 with the onboard computer IP.
 6: Run
    screen
 7: Run
    roscore
 8: Press
    Ctrl+A+C
 9: Run
    roslaunch mavros px4.launch fcu_url:=/dev/ttyUSB0:921600 gcs_url:=udp://@192.168.30.125:14550
    ▷ Replace 192.168.30.125 with the computer IP.
10: Press
    Ctrl+A+C
11: Run
    roslaunch vrpn_client_ros sample.launch server:=192.168.30.105
    ▷ Replace 192.168.30.105 with IP of Optitrack or the other UAV providing the pose estimation of main
    UAV.
12: Press
    Ctrl+A+C
13: Run
    rosrun topic_tools relay /vrpn_client_node/quail/pose /mavros/vision_pose/pose
    ▷ Replace quail with the body name given in the Optitrack.
14: Press
    Ctrl+A+C
15: Run
    rostopic echo /mavros/local_position/pose
    ▷ This is to check if the position of the UAV is displayed. If so, press Ctrl+C to terminate this.
16: Open a new terminal
17: Run
    export ROS_MASTER_URI=http://192.168.30.146:11311
    ▷ Replace 192.168.30.146 with the onboard computer IP.
18: Run
    export ROS_IP=192.168.30.125
    ▷ Replace 192.168.30.125 with the computer IP.
19: Open MATLAB and change the address bar to the to_blf_perception_MRS(with Tello)
20: In MATLAB command window run
    rosinit
21: Open and run the file main.m
    ▷ In line 7, the position of other UAV or obstacle is obtained. In this code, the other UAV is a Tello, whose
    position is measured and published by Optitrack system. So, the name Tello should be changed to the
    body name defined in the Optitrack software. In lines 65 and 66, you can change the positions of initial
```

and final points. Also, in line 68 the last number, you can change the safety radius around the obstacle. Also, the desired height can be changed in line 93. The other UAV is either manually controlled or ASTC can be used. Moreover, if it is an obstacle there is no need for controlling it and only its position is to be measured and and subscribed in line 7.

References

[1] M. Castillo-Lopez, P. Ludivig, S. A. Sajadi-Alamdari, J. L. Sanchez-Lopez, M. A. Olivares-Mendez, and H. Voos, "A real-time approach for chance-constrained motion planning with dynamic obstacles," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3620–3625, 2020.

[2] S. Aggarwal and N. Kumar, "Path planning techniques for unmanned aerial vehicles: A review, solutions, and challenges," *Computer Communications*, vol. 149, pp. 270–299, 2020.

[3] P. Salaris, M. Cognetti, R. Spica, and P. R. Giordano, "Online optimal perception-aware trajectory generation," *IEEE Transactions on Robotics*, vol. 35, no. 6, pp. 1307–1322, 2019.

[4] J. L. Sanchez-Lopez, M. Castillo-Lopez, M. A. Olivares-Mendez, and H. Voos, "Trajectory tracking for aerial robots: an optimization-based planning and control approach," *Journal of Intelligent & Robotic Systems*, vol. 100, no. 2, pp. 531–574, 2020.

[5] D. Schneider and M. Trapp, "Conditional safety certification of open adaptive systems," *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. 8, no. 2, pp. 1–20, 2013.

[6] D. Schneider, R. Adler, P. Feth, J. Reich, and T. Braun, "Safety in cooperative automated systems,"

[7] S. Kabir, I. Sorokos, K. Aslansefat, Y. Papadopoulos, Y. Gheraibia, J. Reich, M. Saimler, and R. Wei, "A runtime safety analysis concept for open adaptive systems," in *International Symposium on Model-Based Safety and Assessment*, pp. 332–346, Springer, 2019.

[8] J. I. Aizpurua, V. M. Catterson, Y. Papadopoulos, F. Chiacchio, and G. Manno, "Improved dynamic dependability assessment through integration with prognostics," *IEEE Transactions on Reliability*, vol. 66, no. 3, pp. 893–913, 2017.

[9] A. Dheedan and Y. Papadopoulos, "Model-based distributed on-line safety monitoring," in *The Third International Conference on Emerging Network Intelligence (EMERGING 2011), Lisbon, Portugal*, pp. 1–7, 2011.

[10] Y. Gheraibia, S. Kabir, K. Aslansefat, I. Sorokos, and Y. Papadopoulos, "Safety+ ai: a novel approach to update safety models using artificial intelligence," *IEEE Access*, vol. 7, pp. 135855–135869, 2019.

[11] S. Kabir, M. Walker, and Y. Papadopoulos, "Dynamic system safety analysis in hip-hops with petri nets and bayesian networks," *Safety science*, vol. 105, pp. 55–70, 2018.

[12] G. Costante, C. Forster, J. Delmerico, P. Valigi, and D. Scaramuzza, "Perception-aware path planning," *arXiv preprint arXiv:1605.04151*, 2016.

[13] J. Müller and G. S. Sukhatme, "Risk-aware trajectory generation with application to safe quadrotor landing," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3642–3648, IEEE, 2014.

[14] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *2011 IEEE international conference on robotics and automation*, pp. 2520–2525, IEEE, 2011.

[15] C. Richter, A. Bry, and N. Roy, "Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments," in *Robotics research*, pp. 649–666, Springer, 2016.

[16] J. Chen, K. Su, and S. Shen, "Real-time safe trajectory generation for quadrotor flight in cluttered environments," in *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 1678–1685, IEEE, 2015.

[17] F. Gao, W. Wu, W. Gao, and S. Shen, “Flying on point clouds: Online trajectory generation and autonomous navigation for quadrotors in cluttered environments,” *Journal of Field Robotics*, vol. 36, no. 4, pp. 710–733, 2019.

[18] S. Liu, M. Watterson, K. Mohta, K. Sun, S. Bhattacharya, C. J. Taylor, and V. Kumar, “Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-d complex environments,” *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1688–1695, 2017.

[19] F. Gao, W. Wu, Y. Lin, and S. Shen, “Online safe trajectory generation for quadrotors using fast marching method and bernstein basis polynomial,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 344–351, IEEE, 2018.

[20] W. Ding, W. Gao, K. Wang, and S. Shen, “An efficient b-spline-based kinodynamic replanning framework for quadrotors,” *IEEE Transactions on Robotics*, vol. 35, no. 6, pp. 1287–1306, 2019.

[21] J. Tordesillas, B. T. Lopez, M. Everett, and J. P. How, “Faster: Fast and safe trajectory planner for navigation in unknown environments,” *IEEE Transactions on Robotics*, vol. 38, no. 2, pp. 922–938, 2021.

[22] W. Ding, W. Gao, K. Wang, and S. Shen, “Trajectory replanning for quadrotors using kinodynamic search and elastic optimization,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7595–7602, IEEE, 2018.

[23] M. Zucker, N. Ratliff, A. D. Dragan, M. Pivtoraiko, M. Klingensmith, C. M. Dellin, J. A. Bagnell, and S. S. Srinivasa, “Chomp: Covariant hamiltonian optimization for motion planning,” *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1164–1193, 2013.

[24] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, “Stomp: Stochastic trajectory optimization for motion planning,” in *2011 IEEE international conference on robotics and automation*, pp. 4569–4574, IEEE, 2011.

[25] H. Oleynikova, M. Burri, Z. Taylor, J. Nieto, R. Siegwart, and E. Galceran, “Continuous-time trajectory optimization for online uav replanning,” in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5332–5339, IEEE, 2016.

[26] F. Gao, Y. Lin, and S. Shen, “Gradient-based online safe trajectory generation for quadrotor flight in complex environments,” in *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 3681–3688, IEEE, 2017.

[27] V. Usenko, L. Von Stumberg, A. Pangercic, and D. Cremers, “Real-time trajectory replanning for mavs using uniform b-splines and a 3d circular buffer,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 215–222, IEEE, 2017.

[28] B. Zhou, F. Gao, J. Pan, and S. Shen, “Robust real-time uav replanning using guided gradient-based optimization and topological paths,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1208–1214, IEEE, 2020.

[29] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, “Chomp: Gradient optimization techniques for efficient motion planning,” in *2009 IEEE International Conference on Robotics and Automation*, pp. 489–494, IEEE, 2009.

[30] X. Zhou, J. Zhu, H. Zhou, C. Xu, and F. Gao, “Ego-swarm: A fully autonomous and decentralized quadrotor swarm system in cluttered environments,” in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4101–4107, IEEE, 2021.

[31] X. Zhou, Z. Wang, H. Ye, C. Xu, and F. Gao, "Ego-planner: An esdf-free gradient-based local planner for quadrotors," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 478–485, 2020.

[32] B. Zhou, F. Gao, L. Wang, C. Liu, and S. Shen, "Robust and efficient quadrotor trajectory generation for fast autonomous flight," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3529–3536, 2019.

[33] F. Gao, L. Wang, B. Zhou, X. Zhou, J. Pan, and S. Shen, "Teach-repeat-replan: A complete and robust system for aggressive flight in complex environments," *IEEE Transactions on Robotics*, vol. 36, no. 5, pp. 1526–1545, 2020.

[34] E. Schmitzberger, J.-L. Bouchet, M. Dufaut, D. Wolf, and R. Husson, "Capture of homotopy classes with probabilistic road map," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3, pp. 2317–2322, IEEE, 2002.

[35] L. Jaillet and T. Siméon, "Path deformation roadmaps: Compact graphs with useful cycles for motion planning," *The International Journal of Robotics Research*, vol. 27, no. 11-12, pp. 1175–1188, 2008.

[36] S. Bhattacharya, "Search-based path planning with homotopy class constraints," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 24, pp. 1230–1237, 2010.

[37] C. Rösmann, F. Hoffmann, and T. Bertram, "Integrated online trajectory planning and optimization in distinctive topologies," *Robotics and Autonomous Systems*, vol. 88, pp. 142–153, 2017.

[38] C. Rösmann, W. Feiten, T. Wösch, F. Hoffmann, and T. Bertram, "Trajectory modification considering dynamic constraints of autonomous robots," in *ROBOTIK 2012; 7th German Conference on Robotics*, pp. 1–6, VDE, 2012.

[39] C. Rösmann, F. Hoffmann, and T. Bertram, "Planning of multiple robot trajectories in distinctive topologies," in *2015 European Conference on Mobile Robots (ECMR)*, pp. 1–6, IEEE, 2015.

[40] H. Oleynikova, Z. Taylor, R. Siegwart, and J. Nieto, "Sparse 3d topological graphs for micro-aerial vehicle planning," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1–9, IEEE, 2018.

[41] F. Blochliger, M. Fehr, M. Dymczyk, T. Schneider, and R. Siegwart, "Topomap: Topological mapping and navigation based on visual slam maps," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3818–3825, IEEE, 2018.

[42] B. Zhou, J. Pan, F. Gao, and S. Shen, "Raptor: Robust and perception-aware trajectory replanning for quadrotor fast flight," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1992–2009, 2021.

[43] J. Alonso-Mora, T. Naegeli, R. Siegwart, and P. Beardsley, "Collision avoidance for aerial vehicles in multi-agent scenarios," *Autonomous Robots*, vol. 39, no. 1, pp. 101–121, 2015.

[44] D. H. Shim, H. J. Kim, and S. Sastry, "Decentralized nonlinear model predictive control of multiple flying robots," in *42nd IEEE International Conference on Decision and Control (IEEE Cat. No. 03CH37475)*, vol. 4, pp. 3621–3626, IEEE, 2003.

[45] T. Nägeli, L. Meier, A. Domahidi, J. Alonso-Mora, and O. Hilliges, "Real-time planning for automated multi-view drone cinematography," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4, pp. 1–10, 2017.

[46] B. Gopalakrishnan, A. K. Singh, M. Kaushik, K. M. Krishna, and D. Manocha, "Chance constraint based multi agent navigation under uncertainty," in *Proceedings of the Advances in Robotics*, pp. 1–6, 2017.

[47] M. Kamel, J. Alonso-Mora, R. Siegwart, and J. Nieto, "Robust collision avoidance for multiple micro aerial vehicles using nonlinear model predictive control," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 236–243, IEEE, 2017.

[48] H. Zhu and J. Alonso-Mora, "Chance-constrained collision avoidance for mavs in dynamic environments," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 776–783, 2019.

[49] D. Lyons, J.-P. Calliess, and U. D. Hanebeck, "Chance constrained model predictive control for multi-agent systems with coupling constraints," in *2012 American control conference (ACC)*, pp. 1223–1230, IEEE, 2012.

[50] A. Gray, Y. Gao, J. K. Hedrick, and F. Borrelli, "Robust predictive control for semi-autonomous vehicles with an uncertain driver model," in *2013 IEEE intelligent vehicles symposium (IV)*, pp. 208–213, IEEE, 2013.

[51] S. L. Herbert, M. Chen, S. Han, S. Bansal, J. F. Fisac, and C. J. Tomlin, "Fastrack: A modular framework for fast and guaranteed safe motion planning," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pp. 1517–1522, IEEE, 2017.

[52] A. Lee, Y. Duan, S. Patil, J. Schulman, Z. McCarthy, J. Van Den Berg, K. Goldberg, and P. Abbeel, "Sigma hulls for gaussian belief space planning for imprecise articulated robots amid obstacles," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5660–5667, IEEE, 2013.

[53] C. Park, J. S. Park, and D. Manocha, "Fast and bounded probabilistic collision detection for high-dof trajectory planning in dynamic environments," *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 3, pp. 980–991, 2018.

[54] J. Hardy and M. Campbell, "Contingency planning over probabilistic obstacle predictions for autonomous road vehicles," *IEEE Transactions on Robotics*, vol. 29, no. 4, pp. 913–929, 2013.

[55] N. E. Du Toit and J. W. Burdick, "Probabilistic collision checking with chance constraints," *IEEE Transactions on Robotics*, vol. 27, no. 4, pp. 809–815, 2011.

[56] L. Blackmore, M. Ono, A. Bektassov, and B. C. Williams, "A probabilistic particle-control approximation of chance-constrained stochastic predictive control," *IEEE transactions on Robotics*, vol. 26, no. 3, pp. 502–517, 2010.

[57] L. Blackmore, M. Ono, and B. C. Williams, "Chance-constrained optimal path planning with obstacles," *IEEE Transactions on Robotics*, vol. 27, no. 6, pp. 1080–1094, 2011.

[58] D. Morgan, G. P. Subramanian, S.-J. Chung, and F. Y. Hadaegh, "Swarm assignment and trajectory optimization using variable-swarm, distributed auction assignment and sequential convex programming," *The International Journal of Robotics Research*, vol. 35, no. 10, pp. 1261–1285, 2016.

[59] S. Liu, M. Watterson, S. Tang, and V. Kumar, "High speed navigation for quadrotors with limited onboard sensing," in *2016 IEEE international conference on robotics and automation (ICRA)*, pp. 1484–1491, IEEE, 2016.

[60] B. T. Lopez and J. P. How, "Aggressive 3-d collision avoidance for high-speed navigation.," in *ICRA*, pp. 5759–5765, 2017.

[61] J. Tordesillas, B. T. Lopez, and J. P. How, "Faster: Fast and safe trajectory planner for flights in unknown environments," in *2019 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 1934–1940, IEEE, 2019.

[62] V. Murali, I. Spasojevic, W. Guerra, and S. Karaman, "Perception-aware trajectory generation for aggressive quadrotor flight using differential flatness," in *2019 American Control Conference (ACC)*, pp. 3936–3943, IEEE, 2019.

[63] D. Falanga, P. Foehn, P. Lu, and D. Scaramuzza, "Pampc: Perception-aware model predictive control for quadrotors," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1–8, IEEE, 2018.

[64] C. Richter and N. Roy, "Learning to plan for visibility in navigation of unknown environments," in *International Symposium on Experimental Robotics*, pp. 387–398, Springer, 2016.

[65] E. Heiden, K. Hausman, G. S. Sukhatme, and A.-a. Agha-mohammadi, "Planning high-speed safe trajectories in confidence-rich maps," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2880–2886, IEEE, 2017.

[66] H. Oleynikova, Z. Taylor, R. Siegwart, and J. Nieto, "Safe local exploration for replanning in cluttered unknown environments for microaerial vehicles," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1474–1481, 2018.

[67] B. Penin, R. Spica, P. R. Giordano, and F. Chaumette, "Vision-based minimum-time trajectory generation for a quadrotor uav," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6199–6206, IEEE, 2017.

[68] R. Spica, P. Robuffo Giordano, and F. Chaumette, "Coupling active depth estimation and visual servoing via a large projection operator," *The International Journal of Robotics Research*, vol. 36, no. 11, pp. 1177–1194, 2017.

[69] G. Costante, J. Delmerico, M. Werlberger, P. Valigi, and D. Scaramuzza, "Exploiting photometric information for planning under uncertainty," in *Robotics research*, pp. 107–124, Springer, 2018.

[70] C. Forster, M. Pizzoli, and D. Scaramuzza, "Appearance-based active, monocular, dense reconstruction for micro aerial vehicles," 2014.

[71] M. Sheckells, G. Garimella, and M. Kobilarov, "Optimal visual servoing for differentially flat underactuated systems," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5541–5548, IEEE, 2016.

[72] C. Potena, D. Nardi, and A. Pretto, "Effective target aware visual navigation for uavs," in *2017 European Conference on Mobile Robots (ECMR)*, pp. 1–7, IEEE, 2017.

[73] G. Elnagar, M. A. Kazemi, and M. Razzaghi, "The pseudospectral legendre method for discretizing optimal control problems," *IEEE Transactions on Automatic Control*, vol. 40, no. 10, pp. 1793–1796, 1995.

[74] R. H. Byrd, M. E. Hribar, and J. Nocedal, "An interior point algorithm for large-scale nonlinear programming," *SIAM Journal on Optimization*, vol. 9, no. 4, pp. 877–900, 1999.

[75] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, no. 1, pp. 25–57, 2006.

[76] A. Safaei and M. N. Mahyuddin, "Optimal model-free control for a generic mimo nonlinear system with application to autonomous mobile robots," *Int. J. Adapt. Control Signal Process.*, vol. 32, no. 6, pp. 792–815, 2018.

[77] A. Safaei, "Adaptive relative velocity estimation algorithm for autonomous mobile robots using the measurements on acceleration and relative distance," *Int. J. Adapt. Control Signal Process.*, vol. 34, no. 3, pp. 372–388, 2020.

[78] J. Q. Cui, S. Lai, X. Dong, and B. M. Chen, "Autonomous navigation of uav in foliage environment," *J. Intell. Robot. Syst.*, vol. 84, no. 1, pp. 259–276, 2016.

[79] L. Besnard, Y. B. Shtessel, and B. Landrum, "Quadrotor vehicle control via sliding mode controller driven by sliding mode disturbance observer," *J. Franklin Inst.*, vol. 349, no. 2, pp. 658–684, 2012.

[80] O. Mofid and S. Mobayen, "Adaptive sliding mode control for finite-time stability of quad-rotor uavs with parametric uncertainties," *ISA Trans.*, vol. 72, pp. 1–14, 2018.

[81] B. Tian, L. Liu, H. Lu, Z. Zuo, Q. Zong, and Y. Zhang, "Multivariable finite time attitude control for quadrotor uav: Theory and experimentation," *IEEE Trans. Indust. Elect.*, vol. 65, no. 3, pp. 2567–2577, 2017.

[82] O. Mofid, S. Mobayen, C. Zhang, and B. Esakki, "Desired tracking of delayed quadrotor uav under model uncertainty and wind disturbance using adaptive super-twisting terminal sliding mode control," *ISA transactions*, vol. 123, pp. 455–471, 2022.

[83] V. I. Utkin and A. S. Poznyak, "Adaptive sliding mode control with application to super-twist algorithm: Equivalent control method," *Automatica*, vol. 49, no. 1, pp. 39–47, 2013.

[84] J. A. Moreno and M. Osorio, "Strict lyapunov functions for the super-twisting algorithm," *IEEE Trans. Automatic Control*, vol. 57, no. 4, pp. 1035–1040, 2012.

[85] V. I. Utkin, "Scope of the theory of sliding modes," in *Sliding modes in control and optimization*, pp. 1–11, Springer, 1992.

[86] C. Edwards and Y. B. Shtessel, "Adaptive continuous higher order sliding mode control," *Automatica*, vol. 65, pp. 183–190, 2016.

[87] H. K. Khalil, "Nonlinear systems third edition," *Patience Hall*, vol. 115, 2002.

[88] C. Edwards and Y. Shtessel, "Adaptive dual layer second-order sliding mode control and observation," in *American Control Conf. (ACC)*, pp. 5853–5858, IEEE, 2015.

[89] A. Chalanga, S. Kamal, L. M. Fridman, B. Bandyopadhyay, and J. A. Moreno, "Implementation of super-twisting control: Super-twisting and higher order sliding-mode observer-based approaches," *IEEE Trans. Indust. Elect.*, vol. 63, no. 6, pp. 3677–3685, 2016.

[90] L. Meier, D. Honegger, and M. Pollefeys, "Px4: A node-based multithreaded open source robotics framework for deeply embedded platforms," in *2015 IEEE Inter. Conf. Robotics Automation (ICRA)*, pp. 6235–6240, IEEE, 2015.

[91] M. Faessler, A. Franchi, and D. Scaramuzza, "Differential flatness of quadrotor dynamics subject to rotor drag for accurate tracking of high-speed trajectories," *IEEE Robot. Autom. Lett.*, vol. 3, no. 2, pp. 620–626, 2017.