Robotics and Autonomous Systems 121 (2019) 103270

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

Learning reciprocal actions for cooperative collision avoidance in quadrotor unmanned aerial vehicles

Amir Behjat[1], Steve Paul[2], Souma Chowdhury[3,*]

Department of Mechanical and Aerospace Engineering, University at Buffalo, Buffalo, NY 14260, United States of America

ARTICLE INFO

Article history:
Received 9 March 2019
Received in revised form 26 July 2019
Accepted 28 August 2019
Available online 4 September 2019

Keywords:
Bio-inspired
Collision avoidance
Learning
Optimization
Unmanned Aerial Vehicle (UAV)

ABSTRACT

The ability to avoid collisions with each other is one of the fundamental requirements for autonomous unmanned aerial vehicles (UAVs) to be safely integrated into the civilian airspace, and for the viability of multi-UAV operations. This paper introduces a new approach for online cooperative collision avoidance between quadcopters, involving reciprocal maneuvers, i.e., coherent maneuvers without requiring any real-time consensus. Two maneuver strategies are presented, where UAVs respectively change their speed or heading to avoid a collision. A learning-based framework that *trains these reciprocal actions for collision evasion* (called TRACE) is developed. The primary elements of this framework include: 1) designing simulated experiments that cover a variety of UAV–UAV approach scenarios; 2) performing optimization to identify speed/heading change actions that satisfy safety constraints while minimizing the energy cost of the maneuver; and 3) using the offline optimization outcomes to train classifier (via ensemble bagged tree) and function approximation (via neural networks and Kriging) models for respectively selecting and encoding the avoidance actions. Trajectory generation and dynamics/controls are incorporated in the simulation environment used for training and testing. Over 90% accuracy in action prediction and over 95% success in avoiding collisions is observed when the trained models are applied to simulated unseen test scenarios.

1. Introduction

1.1. Autonomous UAVs in the civilian airspace

Small unmanned aerial vehicles (UAVs) are becoming ubiquitous in a wide range of low-speed/low-altitude commercial and humanitarian applications, from precision agriculture to disaster response [1]. While operational-safety concerns with regards to human beings on the ground, tall infrastructure, and other manned aircraft sharing the airspace, are a popular topic of mainstream discussion and technical research on UAVs, it is also important to realize that increasing market growth will cause more UAVs to operate in close proximity of each other. Close coordinated operation of UAVs is also fundamental to the concept of *UAV swarm*, which involves a team of (typically simple) UAVs collaborating in a certain manner to offer collectively-intelligent behavior that entails significantly greater mission effectiveness (e.g., robustness, flexibility, redundancy and scalability [2]) compared to sophisticated stand-alone alternatives. In addition to airspace regulations [3,4], one of the major technical barriers to integrating commercial (especially autonomous) UAVs into the airspace or realizing UAV swarm implementations in practice is the immaturity of *autonomous collision avoidance* techniques that are applicable to a wide spectrum of UAVs in terms of cost and sophistication [5]. Autonomous detect, sense and avoid (DSA) techniques [6] are not only imperative to autonomous UAVs, but are also needed as a fail-safe for remotely piloted UAVs that are commonly used in photography, infrastructure survey, and recreation.

While UAVs sharing the civilian airspace and teamed operation of UAVs call for reliable sense and avoid techniques [6], they also present the opportunity to derive and exploit complementary behavior — i.e., *predefined mutually-coherent or complementary maneuvers* — that seek to guarantee successful collision-avoidance in the case of friendly UAV–UAV encounters. A close analogy would be how traffic rules ensure the safety of interactions between automobiles sharing the roadway system. Such a capability also bears the potential to alleviate the online computing load otherwise associated with independent decision-making. This paper develops and evaluates a new (partly bio-inspired) collision avoidance technique that is founded on this notion of "*complementary maneuvers*", and (in its current form) is applicable to encounters between identical multi-rotor UAVs. The

* Corresponding author.
E-mail address: soumacho@buffalo.edu (S. Chowdhury).

[1] PhD Candidate.
[2] Master Student.
[3] Assistant Professor.

https://doi.org/10.1016/j.robot.2019.103270

remaining portion of this section provides a brief survey of existing collision avoidance techniques pertinent to multi-rotor UAVs, and an overview of the underlying concept and objectives of the research presented here.

1.2. Existing collision avoidance approaches

Existing collision avoidance methods, w.r.t both static [7] and dynamic obstacles [8,9] can be broadly classified [10] into: geometric, optimized trajectory, bearing angle, force field, and Markov decision process approaches. A summary discussion of these approaches is provided below.

Geometric approach: These methods alter the trajectory of both UAVs based on the location and velocity of the ownship and intruder UAV [8]. When a collision is detected (e.g., via Kalman filter based state estimation [11]), each agent is given a priority number, and the one with lower priority chooses a waypoint that is perpendicular to its velocity. Although these approaches require the cooperation of the intruder UAV, it does not postulate optimized rules or schemes that each UAV could follow in order to automatically enable collision avoidance at say a minimal energy cost.

Optimized trajectory approach: Under this category, A* and Dijkstra's algorithms are popularly used to compute collision-free trajectories [12]. These algorithms work well when dealing with stationary obstacles, but their applicability is limited in the case of dynamic obstacles or UAV–UAV collision. In the latter case, a reported alternative formulates the path of an agent as an optimization problem that minimizes a cost function using gradient descent or mixed-integer linear programming [13–15] or optimal control theory [16].

Bearing angle approach: These methods follow a near-continuous feedback system, where cameras are used to estimate the relative angle of the obstacle with respect to the UAV, and collision is prevented by keeping the obstacle image at a desired safe position in the camera's field of view [17,18]. While reliable in execution, this places a significant computing demand on the UAV and energy cost of the avoidance maneuver is typically ignored.

Force field approach: Force-field approaches, inspired by attraction/repulsion in electrical fields, determine collision-free trajectories that also satisfy other constraints [19]. The target waypoint for an agent serves as a source of attraction, while obstacles exert a repulsive force. In some implementations, proximity to another UAV incites a braking force [20], while in other [21], the current states of all agents are used to predict the trajectory over the next T horizon using non-linear model predictive control techniques.

Markov Decision Process (MDP) approach: MDP and more recent Partially Observable MDP (POMDP) [22,23] methods make use of the estimated current state of the ownship with respect to its target and (partial) knowledge of the relative state of the intruder UAV. The Airborne Collision Avoidance System X (ACAS-X) class of algorithms are extended to multi-rotor UAV applications, where actions are guided by the principle of maximizing rewards that are estimated based on the expected post-action state of the UAV and its physical constraints (e.g., maximum acceleration). While POMDPs are potent in providing system-aware optimal and safe actions, the online computing burden can become intractable and performance is highly sensitive to tunable parameters, which in [22] are estimated using a Gaussian process approach.

Recent efforts also present the use of reinforcement learning (RL) and deep learning (DL) to operate on the MDP [24,25] and POMDP [26] formulations of the problem for multi-robot collision avoidance in 3-DoF settings. While the promising performance and extensibility (e.g., to multi-agent settings) of learnt models demonstrated in these efforts motivate our adoption of a learning based mechanism to train light-weight online collision avoidance models, there is not indisputable evidence pointing to whether these existing RL/DL frameworks, in their current form, are applicable to UAV–UAV collision avoidance.

Note that some of these existing methods are designed to serve a more general detect-sense-avoid role, thus not necessarily optimized for cooperative UAV–UAV encounters. The focus of this paper is on *collision avoidance between cooperative multi-rotor UAVs*, with an approach that respects the frugal onboard computing capabilities and very limited endurance of small UAVs [27,28]. With this context, the method developed herein seeks to provide the following benefits: (i) Identify and exploit energy-optimal mutual-coordination opportunities that account for the 6-DoF motion complexities of multirotor systems; (ii) Significantly mitigate the online computing burden; (iii) Alleviate the need for continuous (high rate) communication or sensing throughout the maneuver; and (iv) Allow more flexibility for extension in future into newer capabilities (e.g., 3D maneuvers), by performing capability appropriate design of experiments while using the same underlying framework and constitutive trajectory-planning/optimization models as presented in this paper. Moreover, unlike most recent learning based methods, we seek to provide rigorous mathematical description of collision scenarios and a systematic design of experiments, to automatically derive samples with guaranteed collision occurrence (unless evaded via maneuver).

1.3. Collision avoidance via reciprocal actions: A bio-inspired perspective

In this paper, we seek to develop a one-on-one collision avoidance scheme with the following unique combination of characteristics: (1) maneuvers by the two UAVs are proactive and complementary (or reciprocal) while requiring minimal-to-no information exchange during maneuver; (2) avoidance maneuvers are collectively energy optimal for the two UAVs; and (3) the collision-avoidance scheme is computationally lightweight, thereby enabling quasi-instantaneous online decisions. Here "*reciprocal maneuvers*" imply that the action taken by one UAV relies on a coherent action expected from the other UAV, where their actions collectively (seek to) prevent a collision. This notion of *reciprocal* maneuvers is inspired by a particular behavior of birds in a flock, recently reported in a study on budgerigars [29]. The birds were observed to exhibit mutually-coherent collision-avoiding maneuvers during one-to-one head-on scenarios (to distinguish from formation flight concepts). Schiffner and Perez [29] reported that, for a set of 102 wind tunnel experiments with a pair of birds (from a group of ten) each time approaching towards a head-on collision, the success rate of avoiding collisions was 100%. This was accomplished (in a perceptibly rule-based manner) by both birds veering to the right, above or below their original trajectory while passing the other bird. These rules are likely a result of evolution/learning over generations of such bird species. Similar studies have also been conducted on humans [30], e.g., showing how pedestrians adjust their speed and trajectory when crossing a non-reactive human interferer approaching at different angles and speeds (this being a non-cooperative behavior). Drawing inspiration from the collision avoiding behavior of budgerigars, our research asked the following questions: (1) Can we adapt and extend this *mutually-coherent or reciprocal veering* strategy to avoid collisions between UAVs approaching each other from any angle, $\theta \in (0^\circ, 180^\circ]$; (2) How to enable these reciprocal actions to be energy optimal and translate them into a lightweight form that can be computed quasi-instantaneously onboard small UAVs.

In pursuing the answers to these questions, this paper develops and tests a method that *trains reciprocal actions for collision evasion* (TRACE) between multi-rotor UAVs. The developments presented here are focused on (autonomous) avoidance aspect of DSA; the location and velocity information of each UAV is assumed to be available to the other UAV (through inter-UAV communication or state-of-the-art detect/sense approaches). Note that, given the imperfections associated with standard inter-UAV communication and sensing of other flying objects, uncertainties in the state estimation are expected. Although not explicitly modeled in this paper, such state estimation uncertainties can be handled by TRACE by using a higher safety factor in terms of the minimum separation threshold implemented during optimal maneuver planning (with the exception of cases where the controller fails to follow the trajectory).

The technical objectives of this paper can be summarized as follows:

1. Develop a parametric formulation of mutually-coherent (speed or direction change) actions to avoid inter-UAV collisions, and an approach to optimize these actions in terms of energy expense (subject to flight-dynamics constraints).
2. Develop a learning model that can quasi-instantaneously map the relative state of the two UAVs to their optimum reciprocal actions, whenever a potential collision is identified (this model will be trained by the optimizations, and designed to be executed on board UAVs).
3. Evaluate, through simulated experiments, the energy performance and collision-avoidance success rate of: (i) the optimized actions across a large number of UAV–UAV encounter scenarios and (ii) the online schemes trained thereof.

In addition, we aim to also demonstrate the future extensibility of our collision avoidance concept, and that of the underlying computational framework to implement the concept, to other collision avoidance maneuvers (altitude change maneuvers) and collision scenarios (> two UAV collision scenarios).

The remaining portion of this paper is organized as follows: The next section summarizes the underlying assumptions in our method and introduces a mathematical description of inter-UAV collision scenarios. Section 3 presents the development of the new TRACE method for inter-UAV collision avoidance (including design of experiments, optimization, and model training). This is followed by a section describing the implementation settings. Section 5 then discusses the model training and testing results. The paper ends with concluding remarks in Section 6.

2. Scenario description & assumptions

2.1. The UAV system

The UAV–UAV encounter scenarios and the autonomous UAV system considered here are in part motivated by the broader research in the area of collaborative multi-UAV search [31] and mapping [32] applications (which results in closely sharing the airspace). Particularly, identical quadcopter UAVs are assumed with the key specifications summarized in Table 2. Similar to most other reported work in DSA: (i) the motion of the UAVs are assumed to be in a 2D plane; (ii) a calm flying environment with no gusts is considered; and (iii) only one-on-one collisions are considered. We also assume that each UAV can either communicate its current state (location, speed, and heading direction) with its neighboring UAVs or can accurately sense the state of neighboring UAVs (where, practically, sensing range ≫ minimum separation required). Hence, only deterministic scenarios are considered; and each UAV, based on its current state and the information it has about its peer's state, can predict the possibility and time of collision with the other UAV.

2.2. Inter-UAV collision: Scenarios and detection

If the separation between two UAVs becomes lower than a threshold distant d_{col}, then it is termed as a collision. The value of d_{col} can be regulated based on the desired level of safety and can be defined in terms of the UAV size (e.g., 2 × diameter of UAV). Based on the relative speed and heading angle, the distance between any two UAVs can decrease, increase or remain the same. In the context of collision avoidance, one is interested in the cases where the separation distance decreases with time. In order to predict a collision, the time and distance of minimum separation is estimated using the current state of both the UAVs, as described below.

As shown in Fig. 2, let $P_{A,0}$ and $P_{B,0}$ be the current phase of UAVs A and B respectively at a given time point (t_0), and $V_{A,0}$ and $V_{B,0}$ are their respective velocity vectors. Their angle of approach, ϕ, is then given by:

$$\phi = 180^\circ - \cos^{-1}\left(\frac{\hat{V_{A,0}}.\hat{V_{B,0}}}{|\hat{V_{A,0}}.\hat{V_{B,0}}|}\right) \tag{1}$$

For simplicity of representation, vector notations are not used here onward. We will consider a time horizon (t_H) that is safely greater than the time required for deciding and completing any avoidance maneuvers, ($t_0 \leq t \leq t_H$). Assuming that the UAVs continue to travel along their original path (i.e., without any avoidance action), thus moving with a fixed velocity during this time horizon, the locations of UAVs A and B and the Euclidean distance between them at time $t \geq 0$ are then respectively given by:

$$\begin{aligned} P_A(t) &= P_{A,0} + V_{A,0}t \\ P_B(t) &= P_{B,0} + V_{B,0}t \\ d(t) &= |P_A(t) - P_B(t)| \end{aligned} \tag{2}$$

The time point at which the two UAVs will come closest to each other can then be estimated as:

$$t_{\min} = \underset{t\in[t_0,t_H)}{\arg\min}\, d(t) \tag{3}$$

If the UAVs are monotonically diverging from each other, the solution to Eq. (3) will be $t_{\min} = t_0$. If a solution, $t_H > t_{\min} > t_0$, exists within the time horizon, and $d(t_{min}) < d_{col}$, then a collision event is said to have been detected within the time horizon. In that case, the time point of collision is given by:

$$t_{col} = \underset{t\in(t_0,t_{\min})}{\arg\min}\, t, \quad \text{s.t. } d(t) \leq d_{col} \tag{4}$$

Once a collision has been detected and a subsequent avoidance action has been computed, the corresponding time point is denoted as t_1. In practice, t_1 can be set at any $t_0 < t_1 < t_{col}$, such that $t_1 - t_0$ is safely greater than the time required by the UAV's onboard computing system to detect the collision and decide the altered trajectory to take to evade the collision. For the sake of simplicity of representation, in the remainder of the paper, we set the time point $t_1 = 0$. Since the maneuver cannot start before t_1, note that $V_{A,1} = V_{A,0}$ and $V_{B,1} = V_{B,0}$. Each UAV is then expected to start a collision avoidance maneuver at a designed time, $t_2 : t_1 < t_2 < t_1 + \mu(t_{col} - t_1),\quad 0 < \mu \leq 1$. The value of μ can be decided based on desired safety tolerance.

The variation of the inter-UAV separation for generic collision evasion scenarios is illustrated in Fig. 1, where $t_0 < t_i < t_H,\ \forall\, i = 1,\ldots,5$. In this figure, the violet straight line at the bottom represents the threshold for collision (d_{col}); i.e., if the separation between the UAVs go below that threshold, it is termed as a collision. The blue curve shows the separation distance between two UAVs for a scenario where no collision is predicted within

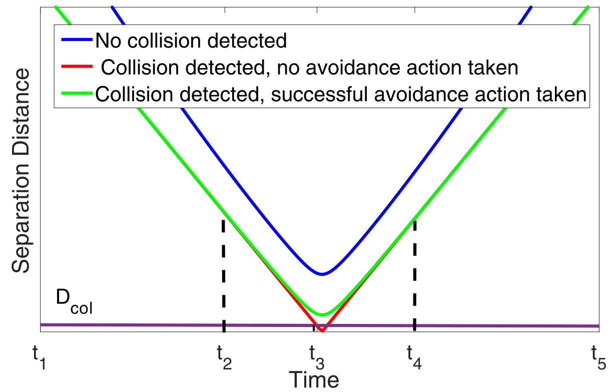


Fig. 1. Separation distance for illustrative cases with no collision, with collision, and successful collision avoidance. Time points are t_1: collision-evasion action decided; t_2: evasion maneuver starts; t_3: symmetry point of evasion maneuver ($t_3 = t_{col}$); t_4: evasion maneuver ends; and $t_5 = t_H$.

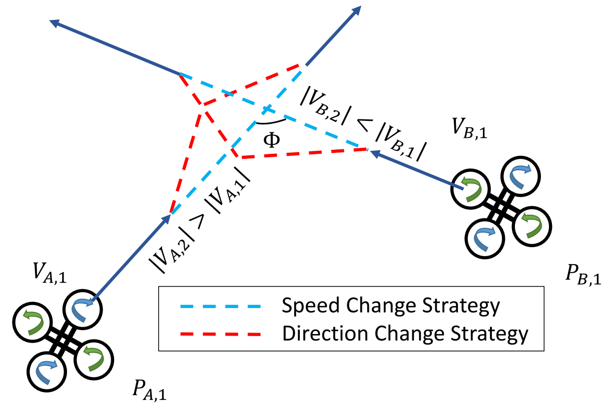


Fig. 2. Representative collision/avoidance scenario, illustrating the two types of avoidance actions (the UAVs actually take smooth trajectories guided by these actions).

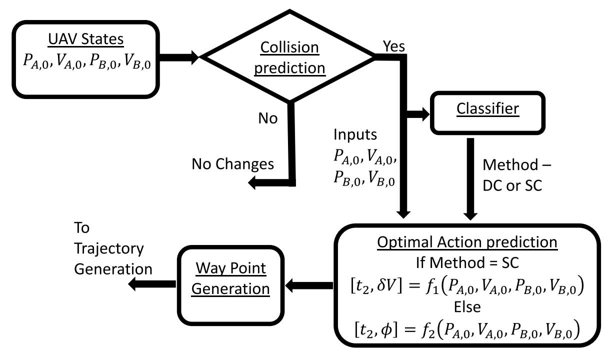


Fig. 3. TRACE: The online collision avoidance process.

the time horizon. The red curve shows the separation distance for a scenario where collision is predicted, and the separation distance curve intersects the collision threshold, d_{col} (yellow line), at time $t_3 = t_{col}$ — i.e., collision occurs. The green curve shows the separation distance for the scenario where collision is predicted (originally similar to the red scenario), but a avoidance action is taken to successfully evade collision. Here 't_2' and 't_4' respectively represent the decided time points at which the collision avoidance maneuver begins and ends.

3. Training Reciprocal Actions for Collision Evasion (TRACE): Framework

3.1. TRACE: Overview

Once a collision is detected, a maneuvering action needs to be decided to avoid collision. In this paper, we consider two different action strategies for collision avoidance: (1) **Direction change**: where both the UAVs veer to the left of their original paths while crossing each other; and (2) **Speed change**: where one UAV slows down and the other speeds up to safely pass each other. While the "direction change" (DC) maneuver is akin to the (head-on) collision avoidance behavior of birds [29], the "speed change" (SC) maneuver was born out of the necessity to deal with scenarios where "direction change" maneuvers do not provide any feasible solutions — this happens particularly when the UAVs approach each other at extreme acute angles (i.e., almost traveling in the same direction). In the case of both strategies, an episode of collision avoidance involves four significant time points, which are described under Fig. 1. A discrete representation of the path changes incurred in these two strategies are illustrated in Fig. 2.

Fig. 3 illustrates how the conceived collision avoidance scheme works. It involves the following four major steps (Fig. 3): (1) *collision prediction*: as described in Section 2.2; (2) *action strategy selection*: a classification process that selects the action strategy (SC or DC) to be used, based on the observed relative pose of the other UAV at detection; (3) *action attribute estimation*: a nonlinear mapping process that estimates the attributes of the action, e.g., deviation angle in DC (further described below) and time when maneuver must start, based on the observed relative pose of the other UAV at detection; and (4) *waypoint generation*: a process that generates intermediate waypoints based on the action attributes. These waypoints are then converted into a smooth flyable trajectory, which is then used by a standard control architecture to execute the avoidance action.

Our stated aim of providing a light-weight collision avoidance approach demands computationally-efficient alternatives to the typically computing-heavy steps. More specifically, selecting the optimum strategy and determining its optimum attributes can be perceived as optimization problem(s) that aim to minimize the additional energy cost associated with the avoidance maneuver, subject to safety constraints (e.g., minimum separation) and physical flight constraints (maximum rated speed). To obviate the computing cost of solving an online optimization, we propose two offline learning approaches to map the relative state of the UAVs at time point t_1 respectively to: (1) an optimum action strategy (DC or SC), and (2) optimum values of the action attributes. A computational framework is developed for implementing these two offline learning approaches.

As shown in Fig. 4, this framework comprises five major components:

- generating a tailored design of experiments (DoE) to create a large variety of one-on-one collision scenarios (training data set);
- optimizing the attributes of the DC/ SC collision avoidance strategies for these DoE scenarios;
- using the scenario-inputs and outcomes of the optimization experiments to train a classifier that selects the optimum action strategy;
- using the scenario-inputs and outcomes of the optimization experiments to train four multi-input-single-output or MISO surrogate models (e.g., based on Kriging or Neural Networks) that can then predict the optimum attributes of the DC/SC actions; and

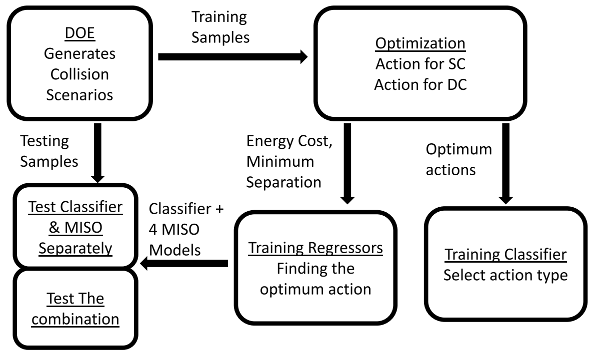


Fig. 4. Computational framework for constructing TRACE.

- testing the four trained models on a new DoE of collision scenarios (test data set) to investigate and compare the selectivity and performance of the two optimized strategies.

The first four components of the framework, along with the models needed for evaluation of any candidate avoidance action (i.e., system dynamics, controls, and trajectory planning models) are described in the following sections.

3.2. Design of Experiments (DoE)

A large set of diverse collision scenarios are designed to capture the possible wide variation in relative states of approaching UAVs where a collision is affirmatively predicted to occur within the time horizon ($t_1 \leq t_{col} \leq t_H$). More specifically, a DoE is performed to generate the initial phases (locations and velocities) of the two approaching UAVs. However, in order to simultaneously ensure that a collision event does exist within the time horizon of the two approaching UAVs and the initial separation is enough to allow feasible collision avoidance, a special constrained sampling approach is needed. Any scenario failing to satisfy the designed constraints (described later) are to be discarded from the learning process. In addition, instead of using the initial poses of the two UAVs to represent the input space of the DoE, the following parameters are used to provide a closed form representation of the relative state of the two UAVs (UAV-A and UAV-B) at time point t_1:

- *Angle of the vector P_{AB} when the separation distance becomes smaller than the threshold* $\rightarrow \theta_{col}$;
- *Heading angle of UAV-A* $\rightarrow \theta_{A,0}$;
- *Heading angle of UAV-B* $\rightarrow \theta_{B,0}$;
- *Speed of UAV-A* $\rightarrow |V_{A,0}|$
- *Speed of UAV-B* $\rightarrow |V_{B,0}|$.

The heading angle and velocity inputs are all specified in terms of a global coordinate system. Table 1 lists the upper and lower bounds for these inputs.

An inverse approach is taken to perform the DoE, starting with points where the UAVs are separated by a distance of d_{col}, the specified separation threshold. The points $P_{A,col}$ and $P_{B,col}$ are respectively the locations of UAV-A and UAV-B at $t = t_{col}$, if the UAVs had continued on their original path with no avoidance maneuver. The angles $\theta_{A,0}$ and $\theta_{B,0}$ are chosen in a manner that guarantees the separation distance, $d(t < t_{col}) > d_{col}$. The initial speeds are then chosen randomly in a range that is practical for UAVs. Fig. 5 illustrates the DoE procedure.

DoE constraints: If constraints were applied post sampling for filtering out UAV–UAV approach scenarios where either no collision is detected or where collision is inevitable (i.e., no feasible avoidance actions exist), the sampling process would have become iterative and time consuming, and possibly biased. Instead, a novel inverse approach is taken, with the practical assumption that UAVs are at a safe distance when a potential collision has been identified within the time horizon. To this end, we set $t_{col} = 5$ s.

Table 1
Range of the input parameters in the DoE.

Parameter	Lower bound	Upper bound
θ_{col}	$0°$	$360°$
$\theta_{A,0}$	$\theta_{col} - 90°$	$\theta_{col} + 90°$
$\theta_{B,0}$	$\theta_{col} + 90°$	$\theta_{col} + 270°$
$\lvert V_{A,0}\rvert$	0.1 m/s	$V_{\max}$
$\lvert V_{B,0}\rvert$	0.1 m/s	$V_{\max}$

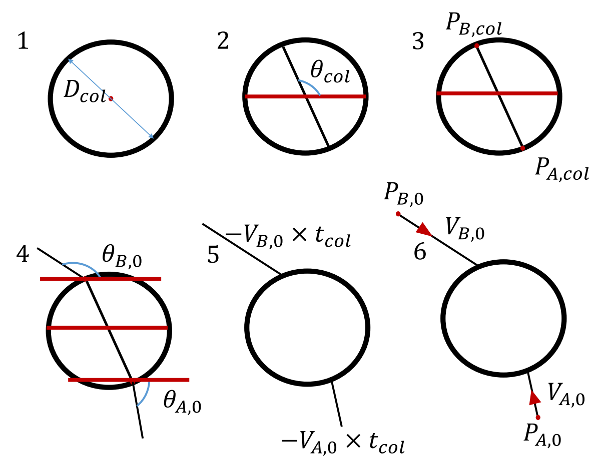


Fig. 5. DoE process: 1: Set global origin: midway between UAVs at $t = t_{col}$. 2: Find separation vector P_{AB} making angle θ_{col} with global X-axis. 3: Determine the UAV positions at t_{col}. 4: Identify UAV heading directions over the time t_0 to t_{col}. 5: Identify the UAV velocities ($V_{A,0}, V_{B,0}$) over this time. 6: Estimate the UAVs' initial positions ($P_{A,0}, P_{B,0}$).

For the DoE, first, uniform random sampling is performed in the space of the five inputs listed in Table 1 ($\theta_{col}, \theta_A, \theta_B, |V_A|, |V_B|$). Optimal Latin Hypercube Sampling (LHS) [33] is used for this purpose. Through a backward in time computation (assuming fixed velocities if no maneuver is taken), the resulting samples are then converted into the corresponding initial poses of the two UAVs ($P_{A,0}, V_{A,0}, P_{B,0}, V_{B,0}$). The conversion process is described below.

The origin (0,0) of the global coordinate system is considered to be at the mid-point of the vector P_{AB} connecting UAV-A and UAV-B at $t = t_{col}$. Then, for each sample scenario, using the input parameter θ_{col} and the prescribed minimum separation threshold of d_{col}, the position of the two UAVs at t_{col} can be expressed as:

$$P_{A,col} = \left(-\frac{d_{col}}{2}\cos(\theta_{col}), \quad -\frac{d_{col}}{2}\sin(\theta_{col})\right) \tag{5}$$

$$P_{B,col} = \left(\frac{d_{col}}{2}\cos(\theta_{col}), \quad \frac{d_{col}}{2}\sin(\theta_{col})\right) \tag{6}$$

Then, assuming the situation where no maneuver action is taken (i.e., the UAVs had continued on their original paths), the initial position of the two UAVs at t_0 are given by:

$$P_{A,0} = P_{A,col} - V_{A,0}.t_{col} \tag{7}$$

$$P_{B,0} = P_{B,col} - V_{B,0}.t_{col} \tag{8}$$

where

$$V_{A,0} = \left(|V_A|\cos(\theta_{A,0}), |V_A|\sin(\theta_{A,0})\right) \tag{9}$$

$$V_{B,0} = \left(|V_B|\cos(\theta_{B,0}), |V_B|\sin(\theta_{B,0})\right) \tag{10}$$

and where θ_A, θ_B, $|V_A|$, and $|V_B|$ are input parameters determined by the sampling.

The following two sub-sections describe how the avoidance actions are parameterized under the SC/DC strategies, and how the optimum action attributes are determined for any given sample scenario — leading to the estimation of the sample outputs.

3.3. Optimal collision avoidance strategies

3.3.1. Speed Change (SC) strategy

In this strategy, we establish a rule whereby the speed of the faster UAV (say UAV-*A*) is increased and the speed of the slower UAV (say UAV-*B*) is decreased, both up to the predicted collision time point (stage-1). Then the reverse action is undertaken by both UAVs to get back to their original speeds (stage-2) such that no net loss/gain in time is incurred over the span of the maneuver (refer Fig. 2). The *mutually coherent two-stage* SC strategy is defined in terms of the average change in speed, δ_V (same in magnitude for both UAVs), and the time point, t_2, when the SC maneuver initiates — together serving as the SC action attributes, to be optimized. The faster UAV accelerates between t_2 and $t_3 = t_{col}$ in a manner such that its average increase in speed is δ_V, and then decelerates back to its original speed by the time point, $t_4 = t_3 + (t_3 - t_2)$. The slower UAV does exactly the opposite during the corresponding time periods. Note that the SC action is symmetric about the time point t_3. The average velocity of the UAVs during the avoidance maneuver, i.e., between time points t_2 to t_3 and t_3 to t_4 are given by:

$$\begin{aligned} \overline{V}_{A,2-3} &= (|V_{A,1}| + \delta_V).\frac{V_{A,1}}{|V_{A,1}|} \\ \overline{V}_{B,2-3} &= (|V_{B,1}| - \delta_V).\frac{V_{B,1}}{|V_{B,1}|} \end{aligned} \tag{11}$$

$$\begin{aligned} \overline{V}_{A,3-4} &= \frac{V_{A,1}(t_4 - t_2) - V_{A,2-3}(t_3 - t_2)}{t_4 - t_3} \\ \overline{V}_{B,3-4} &= \frac{V_{B,1}(t_4 - t_2) - V_{B,2-3}(t_3 - t_2)}{t_4 - t_3} \end{aligned} \tag{12}$$

These average speed estimates will be later used (in Section 3.4.2) to compute the intermediate waypoints defining the altered path associated with the SC maneuver of the two UAVs.

Let us restate the objective of optimal collision avoidance maneuvers in this paper:

- perform a path change that incurs minimum added energy expense compared to the original path,
- while maintaining an inter-UAV separation that is greater than the safety threshold (d_{col}), and
- while satisfying other constraints (e.g., the maximum rated speed of the UAVs).

For the SC maneuver, this optimization problem can be formulated as:

Objective:

$$\min_{t_2, \delta_V} f(t_2, \delta_V, P_{A,0}, P_{B,0}, V_{A,0}, V_{B,0}) \; = E_A + E_B \tag{13}$$

Subject to:

$$\begin{aligned} &g_1 : \max(|V_A(t)|, |V_B(t)|) \leq V_{max}, \;\; \forall \; t_2 \leq t \leq t_4 \\ &g_2 : d(t) \geq d_{col}, \;\; \forall \; t_2 \leq t \leq t_4 \end{aligned} \tag{14}$$

Design variable bounds:

$$\begin{aligned} &t_1 \leq t_2 \leq \mu.t_{col}, \;\; \in \mathbb{R} \\ &0 \leq \delta_V \leq \delta_{V,\max}, \;\; \in \mathbb{R} \\ &\delta_{V,\max} \; = \min\{\, \min\{|V_{A,0}|, |V_{B,0}|\}, \\ &\qquad\qquad V_{\max} - \max\{|V_{A,0}|, |V_{B,0}|\}\} \end{aligned} \tag{15}$$

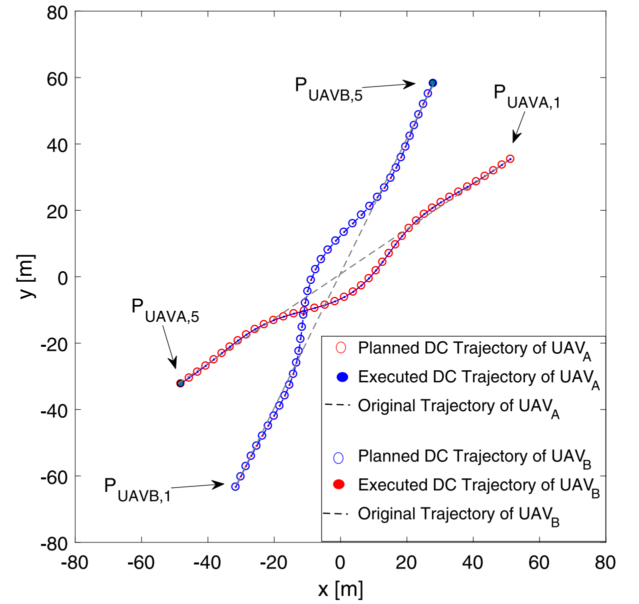


Fig. 6. Direction change (DC) strategy: illustrating planned and executed trajectories (of UAVs A and B) under this strategy.

In Eq. (13), E_A and E_B respectively represent the energy consumption of UAV-*A* and UAV-*B* over the entire time period, t_1 to t_5 (that is inclusive of the period when the avoidance maneuver is active); the energy consumption is given by Eq. (25), described later in Section 3.4.4. The separation distance $d(t)$ used in the optimization (Eq. (14)) is given by Eq. (2). In Eq. (15), μ is a prescribed time safety factor that ensures sufficient time is available for the collision avoidance maneuver. The parameter $V_{\max}$ represents the maximum rated speed of the UAVs.

3.3.2. Direction Change (DC) strategy

In this strategy, we establish a rule whereby each UAV always deviates to the left of its original path at time point t_2. At time point t_{col} (when the UAVs are at their respective extreme deviation point), both UAVs turn right to get back to their original path by time point t_4. A smooth trajectory is planned to execute the maneuver. The approaching UAVs are designated as UAV-*A* and UAV-*B* in a way such that the heading of UAV-*A* is (0° − 180°) rotated w.r.t. the heading of UAV-*B*, where the direction of rotation is counterclockwise. If the designation definition of UAV-*A* and UAV-*B* and the approach angle measurement direction are reversed, this would transform to both UAVs turning right. Fig. 6 provides a representative illustration of a DC maneuver, showing both the planned and the controller executed trajectory of the UAVs. Note that, the "return to original path" constraint is deliberately imposed to allow applicability of our collision avoidance concept to UAV missions where following a planned (often temporally encoded) path is critical to the purpose of the mission, e.g., in data collection missions pertinent to mapping and reconnaissance applications [34,35]. The TRACE concept will readily extend to mission scenarios where this constraint is lifted, potentially allowing even more energy optimal maneuvers (since the feasible space of maneuvers then becomes expanded).

Similar to the SC action strategy, DC can also be perceived as a *mutually coherent two-stage maneuver*. The DC action is defined in terms of the time point, t_2, when the DC maneuver initiates, and the effective change or deviation (going counterclockwise) in the heading direction of the UAVs, angle ϕ (same for both UAVs), incurred between the time point t_2 and the predicted collision time point ($t_3 = t_{col}$). Subsequently, between the times points t_3

and t_4, the UAVs turn back (clockwise) in a manner such that their path/heading merges with the original path/heading by the time point t_4. Note that the DC maneuver is also symmetric about the time point t_3, and is also designed such that no net loss/gain in time is incurred during the time span of the maneuver. The latter condition (for both strategies) is used to ensure that the collision avoidance actions do not affect the overall task schedule of the UAVs in the case of time sensitive missions [32]; this condition can however be relaxed in future implementations given the associated potential risk of actuator saturation.

The original path is a straight line and any deviation results in a longer path to be completed in the same time interval. Therefore both UAVs need to speed up, where their average velocity during the DC maneuver can be expressed as:

$$\begin{aligned} \overline{V}_{A,2-3} &= \frac{1}{\cos\phi} R\, V_{A,1} \\ \overline{V}_{B,2-3} &= \frac{1}{\cos\phi} R\, V_{B,1} \end{aligned} \tag{16}$$

where R is the standard (counterclockwise) rotation matrix in 2D space. The estimated average speed during the avoidance maneuver will be used later on in waypoint planning (Section 3.4.2).

With the same objective and constraints as considered in the SC strategy, the optimization problem for the DC maneuver can be expressed as:

Objective:

$$\min_{t_2,\phi} f(t_2, \phi, P_{A,0}, P_{B,0}, V_{A,0}, V_{B,0}) \ = E_A + E_B \tag{17}$$

Subject to:

$$\begin{aligned} &g_1 : \max(|V_A(t)|, |V_B(t)|) \le V_{max}, \quad t_2 \le t \le t_4 \\ &g_2 : d(t) \ge d_{col}, \quad t_2 \le t \le t_4 \end{aligned} \tag{18}$$

Design variables:

$$\begin{aligned} &t_1 \ \le t_2 \le \mu.t_{col}, \ \in \mathbb{R} \\ &0 \ \le \ \phi \le \phi_{\max}, \quad \in \mathbb{R} \end{aligned} \tag{19}$$

From numerical experiments, the optimal value of the deviation angle (change in heading) was observed to be typically smaller than 30°, and hence $\phi_{\max}$ was set at 30° to helpfully curtail the search space.

3.3.3. Optimization solution approach

It is important to note that, both the SC and DC strategies combine the energy consumption of the two UAVs to form a single objective function, thus not paying any special attention to potential trade-offs between the energy cost incurred by the two UAVs. This is done for simplicity of implementation, given the complexity of the overall TRACE framework proposed in this paper. Future implementations could either pursue a multi-objective optimization approach or impose constraints on the difference in relative additional energy cost incurred by the two UAVs (performing the avoidance maneuvers), in order to facilitate fair distribution of the increased energy expense.

Currently, both the optimization problems, Eqs. (13)–(15) and Eqs. (17)–(19), can be classified as constrained single objective nonlinear optimization problems. From preliminary numerical experiments, the optimization problems were observed to be multi-modal, and the following standard nonlinear optimization algorithms were tested: sequential quadratic programming, genetic algorithms, and swarm optimization algorithms. This allowed us to converge on the usage of Particle Swarm Optimization (PSO) algorithm. In this context, we paid more attention to the quality of the optimum obtained compared to the minor differences in computational cost, since the optimizations are performed offline for each DoE sample. Specifically, we exploited a well-known implementation of the PSO algorithm that offers robustness through explicit diversity preservation [36].

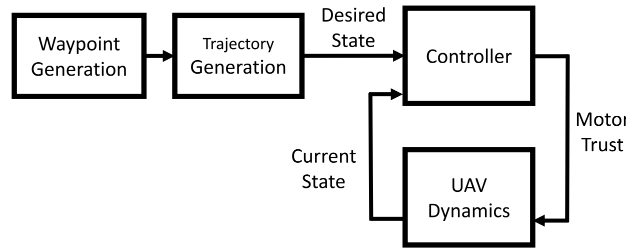


Fig. 7. Trajectory generation and tracking.

3.4. UAV performance modeling

3.4.1. Modeling summary

In order to evaluate the objective function (UAVs' energy consumption) and the constraint functions (separation distance and flight performance constraints) for any given candidate SC or DC action, the following four steps (shown in Fig. 7) are undertaken: (i) waypoint planning, (ii) flyable trajectory generation based on these waypoints, (iii) simulation of the control system seeking to fly the generated trajectory, and (iv) and estimation of energy consumption based on this simulation. These steps are described next.

3.4.2. Waypoint planning

A set of (intermediate) waypoints are generated to define the altered path taken by the UAVs under a collision avoidance action. Given the initial locations of the UAVs ($P_{A,1}$ and $P_{B,1}$) at time point $t_1 = 0$, the waypoint planning process determines the two UAVs' locations at the succeeding time points (t_2 to t_5), as a function of action attributes and assumed constraints — i.e., the UAVs must get back to their original path at t_4 without any net loss/gain in time.

These waypoints are computed in vector form by:

$$\begin{aligned} P_{A,2} &= P_{A,1} + V_{A,1}(t_2 - t_1) \\ P_{B,2} &= P_{B,1} + V_{B,1}(t_2 - t_1) \\ P_{A,3} &= P_{A,2} + \overline{V}_{A,2-3}(t_3 - t_2) \\ P_{B,3} &= P_{B,2} + \overline{V}_{B,2-3}(t_3 - t_2) \\ P_{A,4} &= P_{A,1} + V_{A,1}(t_4 - t_1) \\ P_{B,4} &= P_{B,1} + V_{B,1}(t_4 - t_1) \\ P_{A,5} &= P_{A,1} + V_{A,1}(t_5 - t_1) \\ P_{B,5} &= P_{B,1} + V_{B,1}(t_5 - t_1) \end{aligned} \tag{20}$$

In the case of the speed change (SC) strategy, the velocity $\overline{V}_{A,2-3}$ and $\overline{V}_{B,2-3}$ used in waypoint generation are given by Eq. (11). In the case of the direction change (DC) strategy, the velocity $\overline{V}_{A,2-3}$ and $\overline{V}_{B,2-3}$ are given by Eq. (16). Note that, under the DC action, the designed waypoints, $P_{A,2}$, $P_{A,3}$, and $P_{A,4}$ form an isosceles triangle with a base angle of ϕ (the designed change in heading; an action attribute).

3.4.3. Trajectory generation

The time-stamped intermediate waypoints must be translated into a flyable trajectory that passes through these waypoints. The trajectory can be any single or piece-wise polynomial. If $p(t)$ represents the generic polynomial modeling the UAV's trajectory, and w_k and v_k respectively represent its ith waypoint (P_k, as given by Eq. (20)) and velocity at that waypoint (V_k), then the trajectory polynomial has to satisfy the following conditions:

$$\begin{aligned} p(t_k) &= w_k, \quad k = 1, 2, \ldots, 5 \\ \dot{p}(t_k) &= v_k, \quad k = 1, 5 \end{aligned} \tag{21}$$

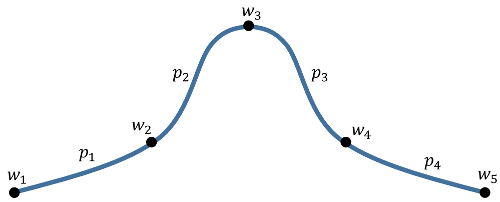


Fig. 8. Piece-wise polynomials connecting waypoints.

Minimum snap trajectory: In this work, we use a piece-wise polynomial trajectory model called the 'Minimum Snap trajectory', introduced by Mellinger and Kumar [37]. "Snap" represents the 4th derivative of the UAV path. As the motor inputs and attitude accelerations (abrupt variations of which are undesirable) of the UAV are proportional to the snap of the path, minimum snap piece-wise polynomials (or splines) prove to be effective in modeling trajectories of multi-rotor UAVs [38]. A minimum snap trajectory provides a smooth path through the way points, while satisfying the conditions expressed in Eq. (21).

Given a set of $n+1$ waypoints with their corresponding time stamps, the minimum snap trajectory generates a piece-wise 7th order polynomial, comprising n segments. Let $w_1, w_2, \ldots, w_{n+1}$ be the set of $n+1$ time-stamped waypoints and $s_1, s_2, \ldots, s_{n+1}$ be their corresponding time points. Each polynomial segment p_i between the consecutive waypoints w_i and w_{i+1} (Fig. 8), can then be expressed as:

$$p_i(t) = \alpha_{i0} + \alpha_{i1}\frac{t-s_i}{s_{i+1}-s_i} + \alpha_{i2}\left(\frac{t-s_i}{s_{i+1}-s_i}\right)^2 + \cdots \\ \ldots + \alpha_{i7}\left(\frac{t-s_i}{s_{i+1}-s_i}\right)^7 \tag{22}$$

Therefore, to create the complete trajectory, we need to determine all the $8n$ coefficients, α_{ij}, where $j = 0, 1, 2, \ldots, 7$ and $i = 1, 2, \ldots, n$. These coefficients can be obtained by solving the following linear system of $8n$ equations:

$$\begin{aligned}
&p_i(s_i) = w_i, \quad \forall\, i = 1, \ldots, n \;\; [n \text{ eqs.}] \\
&p_i(s_{i+1}) = w_{i+1}, \quad \forall\, i = 1, \ldots, n \;\; [n \text{ eqs.}] \\
&\dot{p}_1(s_1) = \dot{p}_n(s_{n+1}) = v_1, \quad [2 \text{ eqs.}] \\
&\dot{p}_i(s_{i+1}) = \dot{p}_{i+1}(s_{i+1}), \quad \forall\, i = 1, \ldots, n-1 \;\; [n-1 \text{ eqs.}] \\
&\ddot{p}_i(s_{i+1}) = \ddot{p}_{i+1}(s_{i+1}), \quad \forall\, i = 1, \ldots, n-1 \;\; [n-1 \text{ eqs.}] \\
&p_i^{(k)}(s_{i+1}) = p_{i+1}^{(k)}(s_{i+1}), \quad \forall\, i = 1, \ldots, n-1 \text{ and} \\
&\qquad\qquad k = 3, \ldots, 6 \;\; [4n-4 \text{ eqs.}] \\
&p_1^{(k)}(s_1) = p_n^{(k)}(s_{n+1}) = 0, \quad \forall\, k = 2, 3 \;\; [4 \text{ eqs.}]
\end{aligned} \tag{23}$$

Note that we will have one system of linear equations (defining the polynomial) each for the time-varying x and y coordinate of the trajectory. Each linear system of $8n$ equations ($A\alpha = b$) can be solved using standard techniques, to yield the coefficient vector α. When implementing Eq. (23) in TRACE, $n = 4$, and v_1 represents the original velocity of the UAV that is taking a collision evasion action.

3.4.4. Dynamics-controls and energy estimation

Here we assume that the quadcopters do not perform complex maneuvers, and hence the yaw is small. This allows us to implement a simple PID controller for trajectory tracking. Since developing a robust controller for the quadcopter is not a focus of this paper, the PID gains used for the simulations are not particularly fine tuned. For the controller, the state of the system consists of its position (X), velocity (V), the Euler angles (Φ, Θ, Ψ) and rate of change of Euler angles ($\dot{\Phi}$, $\dot{\Theta}$, $\dot{\Psi}$), which are typically estimated from the onboard GPS and IMU information. The control system consists of two nested loops. The outer loop performs position control while the inner loop performs attitude control. The inner loop is executed 5 times within the outer loop. The outer loop determines the required total force and the desired Euler angles. The inner loop determines the moments with respect to all the axes. The total force and the moments are fed to the quadcopter dynamics model (adopted from Luukkonen [39]) which computes the thrust for all four rotors and the resulting new state of the UAV. The thrust is in turn used to estimate the power consumption attributed to the four motors.

In this paper, the controller time step is set to 0.05 s for the outer loop and 0.01 s for the inner loop. Hence, during a simulation, as the UAVs follow their trajectory, the thrusts from all four motors of each UAV are estimated at a 0.01 s interval. Subsequently, the power drawn by the four motors is estimated using the following empirical formula adopted from [40]:

$$W_j = 5.8688 \times T_j^{1.4412}, \quad j = 1, 2, 3, 4 \tag{24}$$

Here, T_i and W_i respectively represent the thrust generated (in Newtons) and the power drawn (in Watts) by rotor/motor-i, and the given relationship corresponds to the measured performance of off-the-shelf rotor/motor units used for small quadcopters [40]. The total energy consumption of the UAV between time points t_1 and t_5 can then be expressed as:

$$E = \sum_{i=1}^{4} \int_{t_i}^{t_{i+1}} \sum_{j=1}^{4} P_j(t)dt \tag{25}$$

where the above equation is estimated numerically.

3.5. Training the online TRACE scheme

Steps 2 and 3 of the TRACE method, namely action strategy selection and action attributes estimation, are performed by a classification model and a function approximation model, respectively (as shown in Fig. 4). The classification models are constructed using feed-forward Neural Networks (NN), and both NN regression and Kriging interpolation are explored to perform the function approximation. To these ends, we use the DoE samples (collision scenarios) and sample outputs (corresponding optimized actions) given by the offline optimizations. The model construction processes are described next.

3.5.1. Classifier training: Action selection

Selecting the better action strategy, between SC or DC, is posed as a binary classification problem, as shown earlier in Fig. 3. The inputs to this classification model are the relative initial poses of the two UAVs at the time point of collision detection (t_0), which are $P_{A,0}$, $V_{A,0}$, $P_{B,0}$ and $V_{B,0}$ (i.e., a total of 8 inputs).

The label for each training (DoE) sample is assigned by comparing the estimated performance of the corresponding optimized SC action (solution of Eqs. (13) to (15)) and optimized DC action (solution of Eqs. (17) to (15)). The comparison is driven by a constrained dominance principle, which gives feasibility preference over objective function value; and in this case can be expressed as:

Strategy J dominates strategy K, iff:

- Case I: $d(t)\big|_J^{\min}$ & $d(t)\big|_K^{\min} > d_{col}$ and $(E_A+E_B)|_J < (E_A+E_B)|_K$, OR
- Case II: $d(t)\big|_J^{\min} > d_{col} > d(t)\big|_K^{\min}$, OR
- Case III: $d_{col} > d(t)\big|_J^{\min} > d(t)\big|_K^{\min}$.

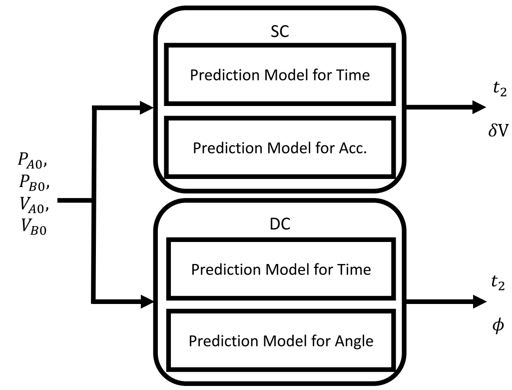


Fig. 9. Inputs/outputs for the TRACE models that decide the action attributes in real-time.

where J and K can be either SC or DC, and $d(t)\big|_J^{\min}$ and $d(t)\big|_K^{\min}$ respectively represent the minimum separation occurring during the maneuver (from t_1 to t_5) when the optimized J and K strategy are used. Case I above represents scenarios where both optimized strategies provide feasible actions, and the more energy-efficient one is chosen. Case II represents scenarios where only one of the strategies provide a feasible action (in terms of required separation distance) and that is the one chosen. Case III represents scenarios where both strategies are unable to produce a feasible action, and the one with the smaller constraint violation is chosen. Further details of the prediction models used for training and the training settings are provided in Section 4.

3.5.2. Function approximation: Action attribute estimation

Four separate multi-input–single-output (MISO) function approximation models (aka. surrogate models) are developed to map collision scenarios to the optimum action attributes defining the SC and DC maneuver, as illustrated in Fig. 9 — let us call them "action attribute estimators". These models are trained over the sample scenarios given by the DoE. For the SC strategy, two surrogate models are trained to predict the time point at which the action ensues (t_2) and the average change in speed (δ_V) to be incurred. For the DC strategy, two surrogate models are trained to predict the time point at which the action ensues (t_2) and the effective change in the heading direction (ϕ) to be incurred. The output labels corresponding to the two SC and the two DC action attributes are given by the outcomes of the offline optimizations over the DoE samples — i.e., SC attributes (t_2^*, δ_V^*) given by solving Eqs. (13)–(15) and DC attributes (t_2^*, ϕ^*) given by solving Eqs. (17)–(19).

We used eight inputs for each MISO model, which represent the initial position and velocity of the two UAVs, i.e., $P_{A,0}$, $V_{A,0}$, $P_{B,0}$, and $V_{B,0}$ (a total of 8 inputs). Input samples containing a reflex angle of approach (i.e., $\delta\theta > 180°$) are transformed into the analogous acute/obtuse angle of approach scenario. Given the natural symmetry of approach scenarios (defined in a 2D plane), every reflex angle of approach case has an equivalent acute/obtuse angle of approach representation, once the designations of UAV-*A* and UAV-*B* are reversed. Note that the state vector comprising the inputs for surrogate modeling is redundant, since a set of 5 state variables or inputs can uniquely define any approach scenario in a 2D plane. Our choice of this 8-sized input vector was driven by the favorable comparison of the corresponding resulting models w.r.t those trained on a vector of 5 inputs.

The "action attribute estimator" models are constructed using Multi Layer Percepteron (feedforward) Neural Networks and Kriging (a Gaussian Process implementation). To train these models, the COSMOS framework is used [41], which allows automated model and kernel selection and automated hyper-parameter tuning. To this end, COSMOS uses a predictive K-fold cross-validation implementation [42]. The selected function approximation models are reported in Section 4.

It is important to note that the offline trained "action attribute estimator" models in TRACE operate in an open loop fashion (Fig. 3), without seeking to perform any online adaptation. In its current form, uncertainties, e.g., due to sensing/state-estimation inaccuracies and environmental factors such as gusts, are not accounted for. Stated otherwise, each UAV is expected to execute their independently-decided (but reciprocal) actions perfectly, and thus the outcome of their actions is assumed to be deterministic.

Table 2
UAV specifications.

Parameters	Value
Quadcopter UAV structure	Plus (+)
UAV Weight	1 kg
UAV Max speed	15 m/s
UAV Max Thrust	8 × Weight
UAV size (diameter)	0.5 m
Safe separation distance (d_{col})	3 × size (1.5 m)
Detection-to-collision time (t_{col})	5 s
Decision time factor (μ)	0.6

4. TRACE: Implementation settings

4.1. Simulation environment and settings

The physical properties and safety thresholds assumed for the quadcopter UAVs are summarized in Table 2; these are derived based on popular small quadcopter platforms such as the DJI Phantom.

Although the collision threshold is defined as 1.5 m, a safety factor of 2 is used in the optimization by setting the d_{col} in the constraints (Eqs. (14) and (18)) to be 3.0 m. This safety factor is considered to compensate for the uncertainty introduced by modeling the optimal actions (to be implemented online) via a regression.

The computational framework for constructing TRACE (Fig. 4), comprising the DoE, optimization, training of the learning models (for action selection and attribute estimation), and the supporting models (for simulating the dynamics, controls, and trajectory planning) are implemented via MATLAB.

4.2. Optimization and learning settings

The optimization problems for speed change (SC) and direction change (DC) maneuvers consist of a single objective and two constraints. Through preliminary numerical experiments with standard gradient-based (SQP), genetic algorithm (GA) and particle swarm optimization (PSO) solvers, PSO was identified to provide better performance. Specifically, we used a variant of the PSO algorithm called MDPSO [36]. The MDPSO algorithm is run using a particle population size of 20 and a maximum allowed iteration of 50. Default values are used for the other parameters [36], which included a relative termination tolerance of 1e−06 on the objective function. The average time taken for each function evaluation (i.e., simulating the trajectory planning and entire control/dynamics of the maneuver) is ∼4 s, running on a single core of a Intel Core i7 7830, 16GB RAM, CPU. Given the high computing cost of running the sample optimizations, parallel computing capabilities, with ∼372–440 nodes of type Intel Xeon

Table 3
Selected classifier and function approximation models and settings: action selection and attribute estimation.

Classifier parameter	Choice
Total # samples	4271
# Samples for training	3846 (∼90%)
# Samples for testing	425 (∼10%)
Model type	Ensemble bagged tree
Testing error (MSE)	12.4%
Func. Approx. Param.	**Choice**
# Samples for training	3846 (∼90%)
# Samples for testing	425 (∼10%)
DC: t_2: Model type	MLP-NN, 20 hidden neurons
DC: ϕ: Model type,	Kriging, exponential kernel
SC: t_2: Model type	MLP-NN, 3 hidden neurons
SC: δV: Model type	Kriging, exponential kernel

E5645 (12 core) CPU and 48GB RAM, is exploited to evaluate the DoE samples.

For classification (action strategy selection), different built in classifiers in MATLAB are applied, and the one with highest accuracy (lowest mean squared error in testing) is selected. The optimization results obtained with the DC and SC maneuver over each DoE sample is subjected to the dominance principles (Case I to III in Section 3.5.1) to identify the corresponding classification label. For function approximation (action attribute estimation), COSMOS exploits the built-in neural network tools in MATLAB and the DACE Kriging implementation [43]. In this case, the labels are given by the optimized SC/DC action parameters obtained for the DoE samples. The selected model types and related parameter settings for training are summarized in Table 3.

5. TRACE: Results and discussion

5.1. TRACE: Optimization results

The main optimization results on the design of experiments is presented and discussed here. In the next sub-section, we present the formulation and outcomes of the ad hoc optimizations that are performed to demonstrate the extensibility of the TRACE framework to handle altitude change maneuvers and maneuvers under multi-UAV collision scenarios. The same optimization method and settings, as stated in Section 4.2, are used in the main DoE and the additional case study's.

A total of 4271 samples is generated by the DoE, each representing a unique collision scenario. The two strategies combined, the optimization process is able to find feasible solutions in 99.2% of the DoE scenarios. The inability to find a feasible maneuver by either SC/DC optimization in the remaining 0.8% scenarios can be attributed to the failure of the controller in maintaining stable flight for the planned trajectory (corresponding to the obtained best candidate actions for that scenario). The pie charts in Fig. 10 illustrates the success rate of the optimized SC and DC maneuvers in finding feasible solutions (i.e., successful collision avoidance) across various approach angle scenarios. It is readily evident that success rate is strongly dependent on the approach angle of the two UAVs — e.g., at near head-on collision scenarios (135°-180°), SC maneuver becomes ineffective, thus leading to a relatively high 43.3% of cases where only optimized DC maneuvers are successful (DC only in Fig. 10(d)). For the other three angle ranges, success overlap (i.e., where both optimized SC/DC is feasible) remains relatively the same. Overall, the DC maneuver can be said to be more effective across a wider range of training scenarios.

Figs. 11 and 12 respectively show the increase in energy consumption resulting from the optimized SC and DC maneuvers, for different angles of approach between the two UAVs, across the

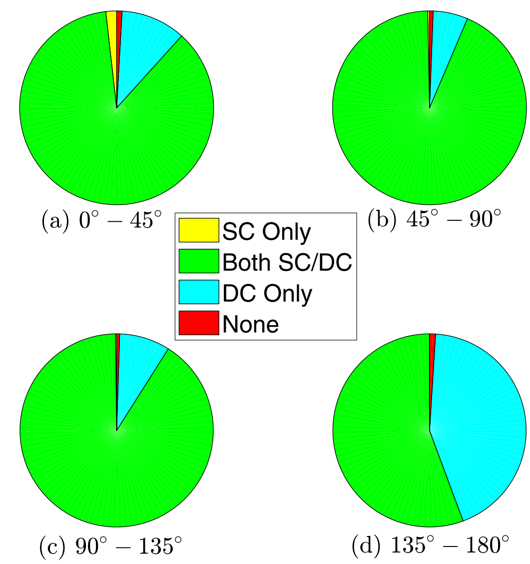


Fig. 10. Fraction of training samples with successful collision avoidance (i.e., min separation distance >3 m) by the optimized SC/DC actions for different range of approach angle, ϕ: (a) [0°–45°], (b) [45°–90°], (c) [90°–135°], (d) [135°–180°].

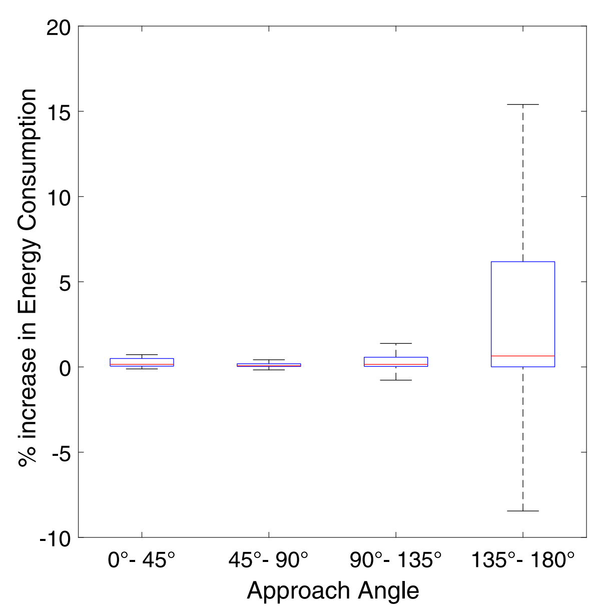


Fig. 11. Speed change (SC) strategy: Increase in energy consumption during maneuver (compared to original path) w.r.t. different approach angles across the DoE.

DoE. Here, the increase in energy is computed w.r.t. the energy cost of both UAVs following the original path without any maneuver. Note that, while a typical quadcopter UAV (as used here) has an energy-optimal forward flight speed, the DoE considers a range of initial speeds (for robustness); thus, there is a small fraction of maneuvers where the net change in energy is negative (if the maneuver brings the UAV closer to their energy-optimal speed). The following is observed from Figs. 11 and 12: (i) for the optimized SC maneuver, the % increase in energy consumption (with median value at 0.5%) is most significant at higher angles of approach (135°–180°), with a large variance observed in near head-on scenarios that demand drastic changes in UAV speeds; and (ii) for the optimized DC maneuver, the % increase in energy consumption (with median value at 0.5%) is most significant at very low angles of approach (0°–45°). Overall, particularly with

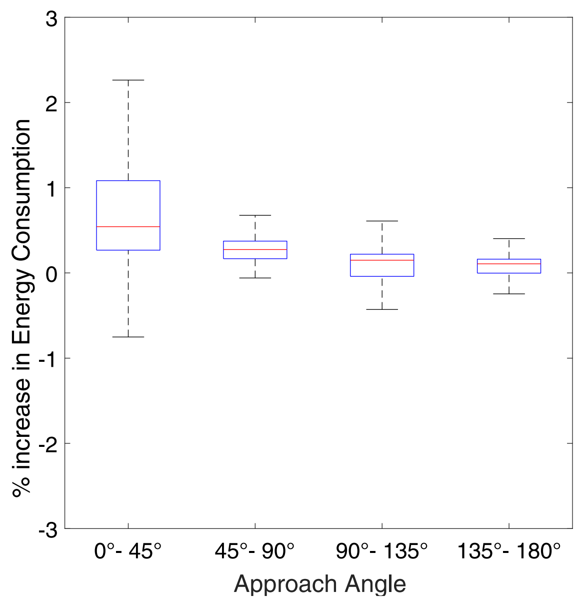


Fig. 12. Direction change (DC) strategy: Increase in energy consumption during maneuver (compared to original path) w.r.t. different approach angles across the DoE.

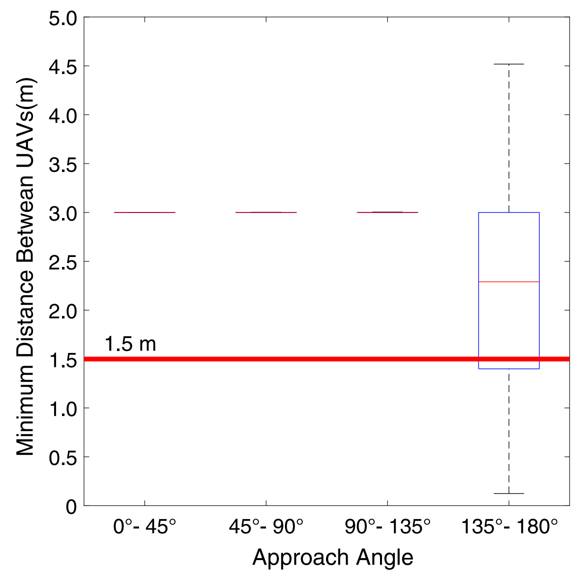


Fig. 13. Speed change (SC) strategy: Minimum separation distance occurring during maneuver w.r.t. different approach angles across the DoE.

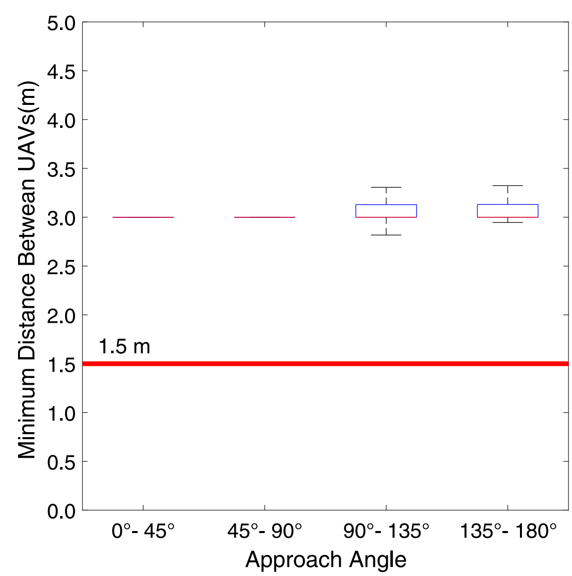


Fig. 14. Direction change (DC) strategy: Minimum separation distance occurring during maneuver w.r.t. different approach angles across the DoE.

the optimized DC maneuver, the % change in energy consumption is desirably low (below 2.5% across all cases).

Figs. 13 and 14 respectively show the minimum distance of separation occurring under the optimized SC and DC maneuvers for different angles of approach across the DoE. The thick red line in both figures represent the actual minimum separation threshold (UAVs getting closer than that threshold is marked as a collision), while a double of that ($d_{col} = 3.0$ m) was set as the threshold in the optimization. Overall, Figs. 13 and 14 illustrate that both SC and DC maneuver effectively avoid collision, and very small variance is observed in this avoidance performance. The only expected exception, as noted earlier, is in the case of SC maneuvers under high approach angle scenarios (135°–180°). The small, but noticeable variance (with some cases maintaining greater than the threshold of 3.0 m minimum separation), observed with the optimized DC maneuver (Fig. 14) can be attributed to scenarios where the energy cost associated with the initial state is significantly deviated from the inherent energy optimal forward flight speed of the UAV.

To analyze the nature of optimal avoidance actions across the different collision scenarios covered by the entire sample set, histograms of the optimum action attributes are provided in Fig. 15. It can be seen from this figure that, for both strategies the maneuver initiation time, t_2, is relatively close to the detection time in most cases, emphasizing the importance of effective sensing/detection methods. This observation can also be partly attributed to the relatively small time-to-collision window of 5s used in this paper. For SC, most scenarios required very small change in speed, typically smaller than 1.0 m/s. For DC, majority of the optimal actions required a deviation in heading angle of about 0° to 10°.

5.2. Demonstrating extensibility of TRACE

5.2.1. Reciprocal altitude change maneuvers

Here, we present a case study to show how the optimization formulation and solution strategy can be readily extended to allow "altitude change" or AC maneuvers as a third option. While "direction change" or DC maneuvers involves temporary path deviations in the horizontal plane (the altitude remaining the same), the AC maneuver involves temporary path deviation in the axial plane (the aircraft's compass direction remaining the same). Both DC and AC are special cases of the more generalized (albeit more complex) planar path deviation maneuver in 3D space. This general maneuver can also be handled with TRACE in the future, by using two angular parameters to define the maneuver, with one parameter encoding the angle of the maneuver plane (say `M`-plane) w.r.t. the horizontal plane and another parameter encoding the UAV's deviation from its original heading direction measured along this `M`-plane.

For the AC maneuver, we use the same minimum separation threshold (d_{col}) settings for ease of illustration. In practice, in the case of AC maneuvers, further careful investigation is needed to consider the aerodynamic downwash effects (of the upper UAV on the lower UAV), which might potentially lead to increasing

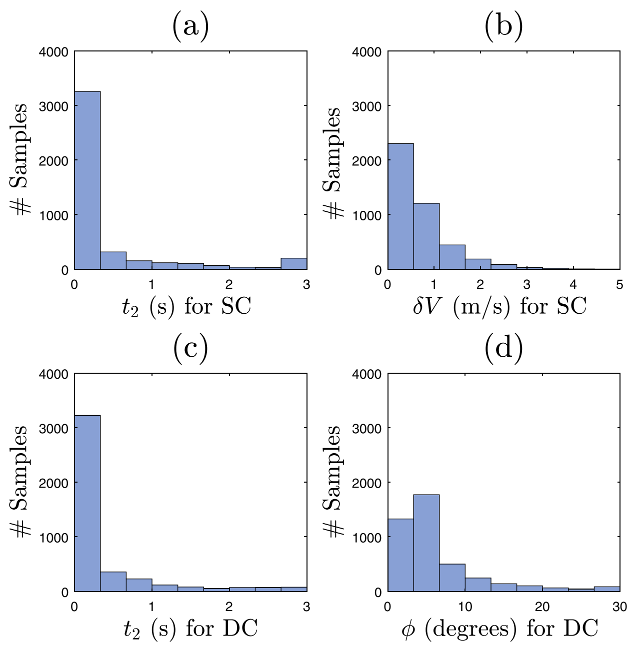


Fig. 15. Optimal action distribution.

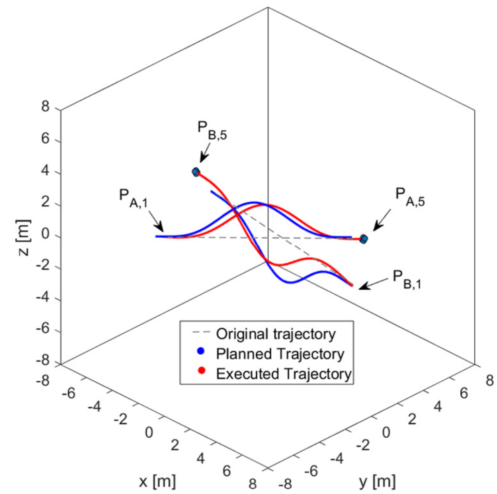


Fig. 16. Collision avoidance with Altitude Change (AC) maneuver by UAVs A and B; gray dashed lines show the original paths; blue and red curves respectively depict the planned and controller-executed trajectories. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

the d_{col} value for AC maneuvers. The overall optimization formulation of the AC maneuver is relatively similar to that of the DC maneuver, and can be stated as:

Objective:

$$\min_{t_2,\psi} f(t_2, \psi, P_{A,0}, P_{B,0}, V_{A,0}, V_{B,0}) = E_A + E_B \tag{26}$$

Subject to:

$$\begin{aligned} &g_1 : \max(|V_A(t)|, |V_B(t)|) \leq V_{max}, \quad t_2 \leq t \leq t_4 \\ &g_2 : d(t) \geq d_{col}, \quad t_2 \leq t \leq t_4 \end{aligned} \tag{27}$$

Design variables:

$$\begin{aligned} &t_1 \ \leq t_2 \leq \mu.t_{col}, \ \in \mathbb{R} \\ &0 \ \leq \ \psi \leq \psi_{\max}, \ \ \in \mathbb{R} \end{aligned} \tag{28}$$

Here ψ is the angular deviation of the UAV from its path in the local X–Z plane. One UAV deviates downward with this angle, while the other deviates upwards with this same angle (just as in DC, decision is conflict free based on the unique designation of the UAVs as A and B). For the ad hoc case study illustrating how the AC maneuvers works in the case of a given collision scenario, we set the upper bound of this deviation angle at $\psi_{\max} = \pi/6$.

The ad hoc optimization is performed for a scenario where the two UAVs are approaching each other at and angle of 146.3°, with UAV-A and UAV-B flying at **1.07** m/s and **1.29** m/s, respectively. Based on their original paths, the UAVs would be passing within **0.56** m of each other. Fig. 16 shows the trajectory of the two UAVs in X–Y and X–Z planes, where the gray dashed lines show the original paths of the two UAVs, and the blue/red circled curves show the (planned/executed) optimized paths under the AC maneuver. Under the optimized AC maneuver, collision is successfully avoided, with the minimum separation staying above **3.06** m throughout the maneuver. It is evident from this illustration that our modeling/optimization approaches are capable of handling altitude change maneuvers, and TRACE's model-training framework can be readily extended in the future to not only include altitude change maneuvers, but also the more general parameterized 3D maneuvers as described earlier. Incorporation of AC maneuvers however might require revisiting the choice of the underlying controller, since AC maneuvers present unique control challenges, as evident from the noticeable deviations between the planned and controller executed trajectory (blue and red curves in Fig. 16).

5.2.2. Multiple UAV collision maneuvers

Unlike 3-DoF ground robots studied in [24,44,45], UAVs typically operate over much larger spaces, even in collaborative search and mapping applications that motivated this work. As a result, the likelihood of more than two UAVs simultaneously arriving at a collision situation is low, with the probability of such occurrences becoming negligible above 3-UAV scenarios; this can be readily checked with simulated numerical experiments. However, to demonstrate that the modeling/optimizations developed in the TRACE framework could still be applicable (with extension) to handle more complex UAV–UAV collision scenarios, a brief case study is provided here showing how to perform optimized collision avoidance with 4 UAVs.

In the case of 4-UAV collision avoidance, the maneuvers, while still reciprocal, need not be symmetric. The space of feasible maneuver solutions (agnostic of the method used to compute the maneuver) significantly decreases as one increases the number of simultaneously-colliding UAVs. Thus, we remove the restriction of all UAVs needing to choose either the SC maneuver or the DC maneuver; i.e., in this case, the maneuvers of the four UAVs can be some combination of SC and DC, depending on what yields the most energy optimal performance subject to the separation constraints. Therefore, to allow greater flexibility, here the optimization parametrization is extended to include 5 design variables for each UAV. Taken together for the four UAVs, the 4-element variable vectors now include the maneuver strategy selections ($\bar{s}$), the times to start the SC or DC maneuvers ($\bar{t}_{2,\mathrm{SC}}$ or $\bar{t}_{2,\mathrm{DC}}$), the angular deviations for the DC maneuver ($\bar{\phi}$), and the speed changes for the SC maneuver ($\bar{\delta}_V$). The expanded optimization problem can be defined as:

Objective:

$$\min_{\bar{s},\bar{t}_{2,\mathrm{SC}},\bar{\delta}_V,\bar{t}_{2,\mathrm{DC}},\bar{\phi}} f(\bar{s}, \bar{t}_{2,\mathrm{SC}}, \bar{\delta}_V, \bar{t}_{2,\mathrm{DC}}, \bar{\phi}, \bar{P}_0, \bar{V}_0) = \sum_{i=1}^{4} E_i \tag{29}$$

Table 4
Multiple UAV optimization setting.

Design variable	Lower	Upper
$t_{2,\mathrm{SC}}$	0 s	3 s
δV_{SC}	$-V_i$ m/s	$V_{\max} - V_i$ m/s
$t_{2,\mathrm{DC}}$	0 s	3 s
ϕ_{DC}	$-\frac{\pi}{6}$ rad	$+\frac{\pi}{6}$ rad
s	{SC, DC}	

Subject to:

$$
\begin{aligned}
&g_1 : |V_i(t)| \leq V_{max}, && \forall\, t_2 \leq t \leq t_4 \\
&g_2 : d_{i,j}(t) \geq d_{col}, && \forall\, t_2 \leq t \leq t_4 \\
& && \forall i, j \in \{1, \ldots, 4\}, i \neq j
\end{aligned} \tag{30}
$$

Design variable bounds:

$$
\begin{array}{rcll}
 & s_i & & \in \{\mathrm{SC}, \mathrm{DC}\} \\
t_1 \leq & t_{2,\mathrm{SC},i} \leq \mu.t_{col}, & & \in \mathbb{R} \\
-V_{0,i} \leq & \delta_{V,i} \leq V_{\max} - V_{0,i}, & & \in \mathbb{R} \\
t_1 \leq & t_{2,\mathrm{DC},i} \leq \mu.t_{col}, & & \in \mathbb{R} \\
\phi_{\max} \leq & \phi_i \leq \phi_{\max}, & & \in \mathbb{R} \\
\forall & i & & \in \{1, \ldots, 4\}
\end{array} \tag{31}
$$

Table 4 lists the design variables bounds for this optimization. Fig. 17 shows the original paths (gray dashed lines) and the trajectory under the optimized maneuver of these 4 UAVs (blue/red curves). It can be observed that while the original paths led to a minimum separation of 0.31 m (thus a collision), the optimized maneuver ensures the minimum separation between all four UAVs stays above 3.03 m (successful collision evasion). UAVs A, B, and C are observed to have used DC maneuvers with different deviation angles, while UAV-D chose a SC maneuver. The total energy consumption for all four UAVs is 1588 J (compared to 1550 J under the original collision path). Therefore, it is conceivable that if needed, the modeling and optimization-based sampling in TRACE can be extended to multi-UAV collision scenarios.

5.3. TRACE: Model training and testing results

The 3846 training samples and labels generated from the corresponding optimizations are used to train the classification model for action strategy selection, and to train the four function approximation models for action attribute estimation. For the maneuver initiation time attribute (t_2), MLP neural networks (NN) were chosen by COSMOS for SC and DC strategies. In contrast, for the average change in speed, δV (an SC attribute), and effective angular change in heading, ϕ (in DC), Kriging models were chosen by COSMOS. The training performance of these models are reported in Table 5. Here, the error of the classifier is given by 10-fold cross-validation, and converted into a relative error measure through scaling using the number of samples. The error of the function approximation models (MLP and Kriging) are given by PEMF [42], a predictive K-fold cross-validation approach. To derive relative absolute error measures (RAE), the absolute values of the function approximation error are normalized using the following prescribed/observed range of these action attributes: (i) $\mu t_{col} = 3.0s$ for t_2 in both SC and DC; (ii) 5.0 m/s for δV in SC; and (iii) $\phi_{\max} = \pi/6$ or 30° for ϕ in DC. The accuracy estimates, reported in Table 5, are then obtained by subtracting the relative error measures (expressed in %) from 100. As seen from Table 5, both the classifier and function approximation models provide reasonable to high accuracy, based on the cross-validation measures.

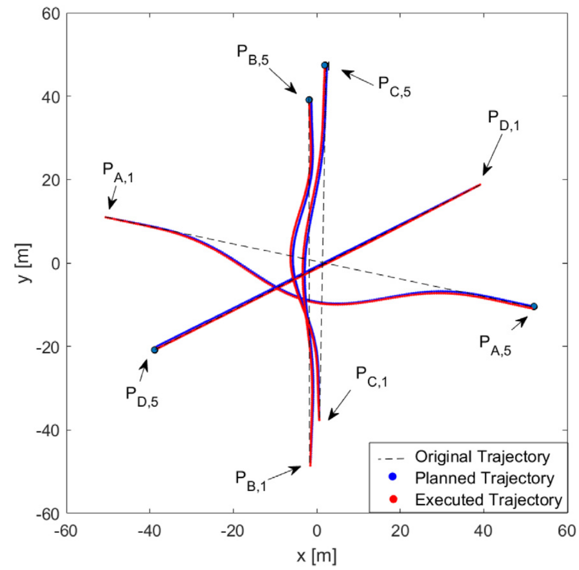


Fig. 17. Collision avoidance for 4 UAV scenario, using SC and DC maneuvers; the UAVs are defined as A, B, C and D; gray dashed lines show the original paths; blue and red curves respectively depict the planned and controller-executed trajectories. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 5
Training results: Action selection (classifier) and attribute estimation (function approx.) model accuracies.

Selection	Classification model	CV accuracy
SC vs. DC	Ensemble bagged tree	87.6%
Attribute	**Func. Approx. Model**	**CV accuracy**
SC, t_2	NN, 3 Hidden neurons	91.6%
SC, δV	Kriging, Exp. Kernel	96.4%
DC, t_2	NN, 20 Hidden neurons	94.9%
DC, ϕ	Kriging, Linear kernels	97.4%

Table 6
Testing results: Action selection (classifier) and attribute estimation (function approx.) model accuracies.

Selection	Classification model	Test accuracy
SC vs. DC	Ensemble bagged Tree	87.5%
Attribute	**Func. Approx. Model**	**Test accuracy**
SC: t_2	NN, 3 Hidden neurons	90.7%
SC: δV	Kriging, Exp. Kernel	97.6%
DC: t_2	NN, 20 Hidden neurons	96.2%
DC: ϕ	Kriging, Linear kernels	94.7%

The performance of the action strategy selection (classifier) and action attribute estimation models is also evaluated on the test set of 425 unseen samples. In this case, the error (or conversely the accuracy) is computed by directly comparing the outputs given by the trained (action strategy selection and action attribute estimation) learning models with that derived from running the full SC/DC action optimization on those test samples. The testing accuracy of the models is reported in Table 6. The same scaling and normalization approach as described earlier (for training accuracy) is used to derive the % accuracy estimates. Table 6 shows that the testing performance of the models is well aligned with that predicted (via cross-validation) during training, and the reasonable accuracy thus observed highlights the computational effectiveness of the TRACE framework.

Further analysis of the classification performance on the test samples is provided in Table 7. The reported mis-classification

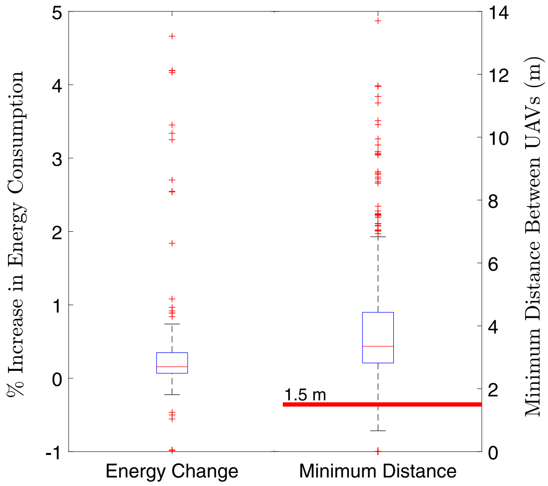


Fig. 18. Change in energy and minimum distance between UAVs under actions taken by the TRACE models over the test scenarios.

rate shows that the classifier is slightly biased towards the DC strategy, likely caused by the disproportionate number of better outcomes given by DC across the training samples.

5.4. TRACE: Performance analysis

To analyze the physical performance of the models trained by TRACE, we look at the two important quantities of interest computed over the test samples. These are the change in energy and the minimum distance of separation resulting under the maneuvers given by the trained TRACE models. Both quantities are computed by executing the control/dynamics simulation in response to the actions decided by the trained TRACE models (for each test scenario). Fig. 18 illustrates these two quantities as boxplots computed over the 425 test scenarios. It can be seen from Fig. 18 that the % change in energy consumption is desirably low, with a median value less than 1%. With regards to the minimum distance of separation between the two cooperating UAVs, the median value is observed (from Fig. 18) to be desirably close to 3.0 m; a small set of samples are demonstrating notably higher separation (likely attributed to opportunistic increase in speed to get close to the optimal operating speed of the UAV). The failure rate, i.e., the fraction of samples failing to avoid collision (i.e., minimum separation $<$1.5 m threshold) is found to be 20/425. This rate, at $\sim$4.7%, remains low, and is only slightly higher than that observed across the training samples (which had a 3.0% failure rate).

Note that (from Fig. 18 and Table 7), a majority of the misclassified test samples still resulted in successful collision avoidance. This is partly attributed to the safety factor of 2 used in setting the minimum distance threshold constraints during the optimizations that generate the model training labels.

Lastly, we measure the average decision-making time taken by the online collision avoidance scheme trained by TRACE (Fig. 3), when implemented on a Intel Core i7 7820HQ CPU/16 GB memory workstation. The average time was found to be 0.032 ms, thereby demonstrating the remarkable efficiency of this approach, and thus suitability for practical implementation onboard UAVs with frugal computing capacities. An illustrative example of the simulated implementation of the TRACE models (showing the flow of information and the resulting trajectories) is provided in Fig. 19 at the end of this paper.

6. Conclusion

In this paper, we proposed a novel cooperative collision avoidance concept for quadrotor UAVs, and developed a framework (called TRACE) to train the models implementing this concept. Here, two approaching UAVs undertake mutually reciprocal maneuvers, in terms of either change in direction (heading) or change in speed. Important considerations, mainly seeking the maneuver to be collectively energy optimal and the requirement for the UAVs to come back to the original path within a given time point, furthers the utility of the proposed concept. A supervised learning approach was taken to train a classifier and multiple function approximation models, that together serve as the computationally lightweight decision system to be used online. The classifier is used to select between the direction change or speed change strategy depending upon their safety/energy-performance combination. The function approximation models are used to predict the action parameters defining the maneuver (e.g., when to start the maneuver and degree of change in heading). An efficient design of experiments (DoE) was performed to generate a set of scenarios where collisions are guaranteed, and a reasonable coverage of potential approach situations (involving two UAVs) is facilitated. This DoE could be useful for other researchers who aim to take an offline learning based approach to collision avoidance between flying robots. The classification/action model labels were generated by running an optimization over each sample scenario given by the DoE.

Training and testing were performed in a simulation environment, with both involving trajectory planning, and implementing a PID controller to fly the trajectory. Over the training scenarios, the optimized DC actions were found to be more effective over a wider range of UAV–UAV approach angles, with the change in energy consumption (i.e., *with maneuver* vs. *without maneuver*) being within 2.5%. Most optimized actions expectedly identified a maneuver initiation time very close to the collision detection time. Ensemble bagged tree was chosen as the classifier, and a model selection framework identified different Kriging and Neural Network models for action prediction. The accuracy of the classifier was found to be $\sim$87.5% in both training and testing, and action models' prediction accuracy (compared to the optimum values) were all found to be greater than 90%, when tested over a set of 425 unseen scenarios. These results highlight the effectiveness of the TRACE framework in training the models that comprise the online collision avoidance system. Further performance analysis over the unseen scenarios resulted in a 95.3% success rate in avoiding collisions.

While a tight time window for detection-to-collision, of only 5 s, was used in the design of experiments to conservatively account for typical sensor range/latency capabilities, sensor noise has not yet been taken into account. In its current form, the TRACE framework considers only deterministic collision scenarios, where pose estimation is assumed to be perfect. Accounting for uncertainties attributed to imperfect pose estimation and wind effects, and the possibility of in-maneuver adaptation, are thus important next steps in this research. These, along with deployment and testing on hardware platforms, would help further establish the effectiveness of this reciprocal collision avoidance concept. Other directions of future research that could advance the applicability of the proposed framework include consideration of non-identical UAVs and allowance of altitude change maneuvers.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Table 7
Results of the (simulated) online collision avoidance tests.

Collision avoidance	# Tests	# SC	% Misclassified as SC	# DC	% Misclassified as DC
Successful ($D \geq 1.5m$)	405/425	100/405	10%	305/405	20%
Failure ($D < 1.5m$)	20/425	2/20	0%	18/20	0%

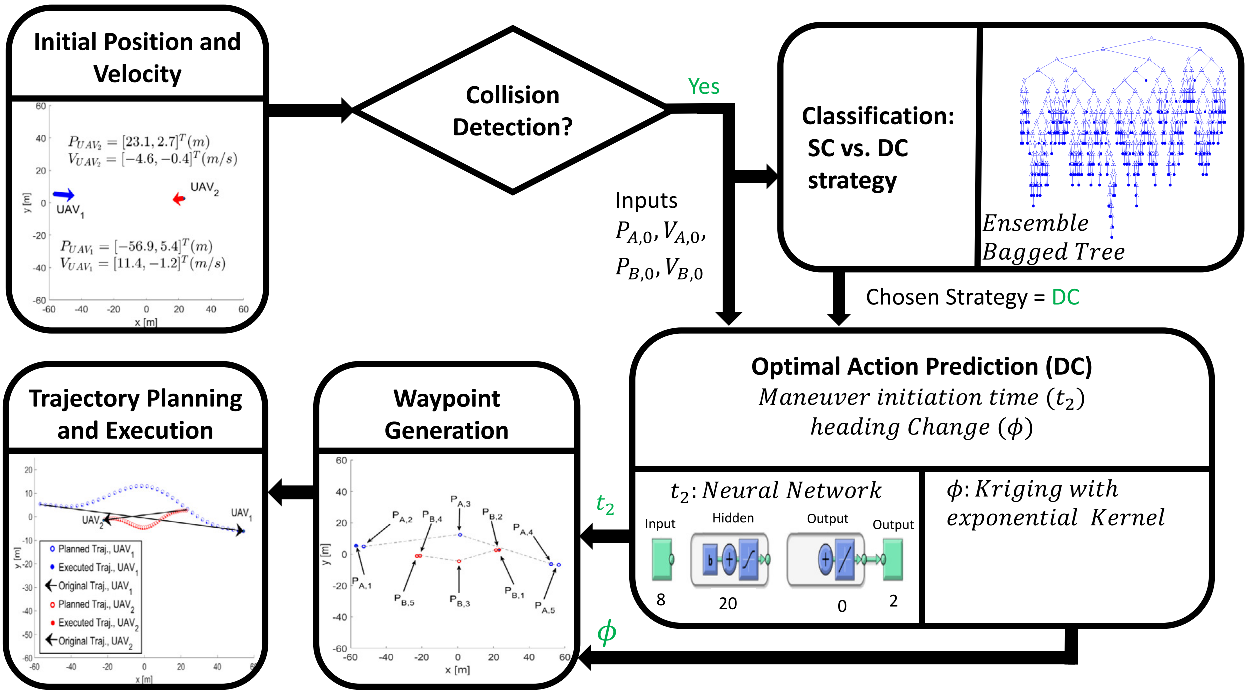


Fig. 19. TRACE models in operation: showing initial pose, decisions taken by each component model of TRACE, flow of information, and resulting trajectory with successful collision avoidance (faster UAV has longer overall trajectory).

Acknowledgment

Support from the DARPA Award HR00111890037 from Physics of AI (PAI) program is gratefully acknowledged.

References

[1] PWC, Global market for commercial applications of drone technology valued at over $127bn, 2016, http://pwc.blogs.com.

[2] D.M. Silva, L.F.F. de Oliveira, M.G. Macedo, C.J. Bastos Filho, On the analysis of a swarm intelligence based coordination model for multiple unmanned aerial vehicles, in: 2012 Brazilian Robotics Symposium and Latin American Robotics Symposium, IEEE, 2012, pp. 208–213.

[3] European Union Aviation Safety Agency, Civil drones (Unmanned aircraft). https://www.easa.europa.eu/easa-and-you/civil-drones-rpas. (Online; Accessed 25 July 2019).

[4] Federal Aviation Administration, Small unmanned aircraft regulations, 2018, https://www.faa.gov/news/fact_sheets/news_story.cfm?newsId=22615. (Online; Accessed 25 July 2019).

[5] T. Frey, 37 critical problems that need to be solved for drone delivery to become viable, 2015, http://www.futuristspeaker.com/business-trends/37-critical-problems-that-need-to-be-solved-for-drone-delivery-to-become-viable/.

[6] S.B. Hottman, K. Hansen, M. Berry, Literature review on detect, sense, and avoid technology for unmanned aircraft systems, 2009.

[7] M. Iacono, A. Sgorbissa, Path following and obstacle avoidance for an autonomous UAV using a depth camera, Robot. Auton. Syst. 106 (2018) 38–46.

[8] S. Tang, V. Kumar, Translating paths into optimal trajectories for safe coordination of teams of dynamic robots, in: Robotics: Science and Systems (RSS), Robotics: Science and Systems (RSS), Workshop on On-line Decision-Making in Multi-robot Coordination, Ann Arbor, Michigan, 2016.

[9] M.H. Tanveer, C.T. Recchiuto, A. Sgorbissa, Analysis of path following and obstacle avoidance for multiple wheeled robots in a shared workspace, Robotica 37 (1) (2019) 80–108.

[10] H. Pham, S.A. Smolka, S.D. Stoller, D. Phan, J. Yang, A survey on unmanned aerial vehicle collision avoidance systems, 2015, arXiv preprint arXiv:1508.07723.

[11] C. Luo, S.I. McClean, G. Parr, L. Teacy, R. De Nardi, UAV position estimation and collision avoidance using the extended kalman filter, IEEE Trans. Veh. Technol. 62 (6) (2013) 2749–2762.

[12] T. Liao, UAV Collision Avoidance using A* Algorithm (M.S. thesis), Auburn University, Auburn, Alabama, 2012.

[13] M. Zhang, J. Song, L. Huang, C. Zhang, Distributed cooperative search with collision avoidance for a team of unmanned aerial vehicles using gradient optimization, J. Aerosp. Eng. 30 (1) (2016) 04016064.

[14] S.K. Gan, R. Fitch, S. Sukkarieh, Real-time decentralized search with inter-agent collision avoidance, in: 2012 IEEE International Conference on Robotics and Automation, IEEE, 2012, pp. 504–510.

[15] A. Richards, J.P. How, Aircraft trajectory planning with collision avoidance using mixed integer linear programming, in: Proceedings of the 2002 American Control Conference (IEEE Cat. No. CH37301), Vol. 3, IEEE, 2002, pp. 1936–1941.

[16] Y. Choi, H. Jimenez, D.N. Mavris, Two-layer obstacle collision avoidance with machine learning for more energy-efficient unmanned aircraft trajectories, Robot. Auton. Syst. 98 (2017) 158–173.

[17] A. Mcfadyen, A. Durand-Petiteville, L. Mejias, Decision strategies for automated visual collision avoidance, in: 2014 International Conference on Unmanned Aircraft Systems (ICUAS), IEEE, 2014, pp. 715–725.

[18] X. Yang, L.M. Alvarez, T. Bruggemann, A 3D collision avoidance strategy for UAVs in a non-cooperative environment, J. Intell. Robot. Syst. 70 (1–4) (2013) 315–327.

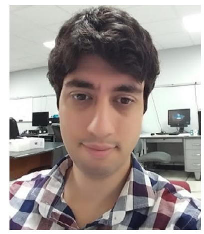
[19] A. Miura, H. Morikawa, M. Mizumachi, Aircraft collision avoidance with potential gradient—ground-based avoidance for horizontal maneuvers, Electron. Commun. Japan 3 78 (10) (1995) 104–114.

[20] D.E. Chang, S.C. Shadden, J.E. Marsden, R. Olfati-Saber, Collision Avoidance for Multiple Agent Systems, IEEE, 2003.

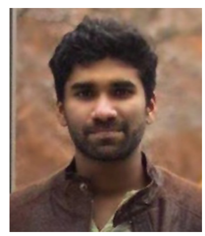
[21] D.H. Shim, H.J. Kim, S. Sastry, Decentralized nonlinear model predictive control of multiple flying robots, in: 42nd IEEE International Conference on Decision and Control (IEEE Cat. No. 03CH37475), Vol. 4, IEEE, 2003, pp. 3621–3626.

[22] E.R. Mueller, M. Kochenderfer, Multi-rotor aircraft collision avoidance using partially observable Markov decision processes, in: AIAA Modeling and Simulation Technologies Conference, 2016, p. 3673.

[23] S. Temizer, M. Kochenderfer, L. Kaelbling, T. Lozano-Pérez, J. Kuchar, Collision avoidance for unmanned aircraft using Markov decision processes, in: AIAA Guidance, Navigation, and Control Conference, 2010, p. 8040.
[24] Y.F. Chen, M. Liu, M. Everett, J.P. How, Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning, in: 2017 IEEE International Conference on Robotics and Automation, ICRA, IEEE, 2017, pp. 285–292.
[25] P. Long, W. Liu, J. Pan, Deep-learned collision avoidance policy for distributed multiagent navigation, IEEE Robot. Autom. Lett. 2 (2) (2017) 656–663.
[26] P. Long, T. Fanl, X. Liao, W. Liu, H. Zhang, J. Pan, Towards optimally decentralized multi-robot collision avoidance via deep reinforcement learning, in: 2018 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2018, pp. 6252–6259.
[27] G. Cai, J. Dias, L. Seneviratne, A survey of small-scale unmanned aerial vehicles: Recent advances and future development trends, Unmanned Syst. 2 (02) (2014) 175–199.
[28] V. Kumar, N. Michael, Opportunities and challenges with autonomous micro aerial vehicles, Int. J. Robot. Res. 31 (11) (2012) 1279–1291.
[29] I. Schiffner, T. Perez, M.V. Srinivasan, Strategies for pre-emptive mid-air collision avoidance in budgerigars, PLoS One 11 (9) (2016) e0162435.
[30] M. Huber, Y.-H. Su, M. Krüger, K. Faschian, S. Glasauer, J. Hermsdörfer, Adjustments of speed and path when avoiding collisions with another pedestrian, PLoS One 9 (2) (2014) e89589.
[31] R.A. Clark, G. Punzo, C.N. MacLeod, G. Dobie, R. Summan, G. Bolton, S.G. Pierce, M. Macdonald, Autonomous and scalable control for remote inspection with multiple aerial vehicles, Robot. Auton. Syst. 87 (2017) 258–268.
[32] P. Odonkor, Z. Ball, S. Chowdhury, Distributed operation of collaborating unmanned aerial vehicles for time-sensitive oil spill mapping, Swarm Evol. Comput. 46 (2019) 52–68.
[33] J.-S. Park, Optimal latin-hypercube designs for computer experiments, J. Statist. Plann. Inference 39 (1) (1994) 95–111.
[34] S.M. Adams, C.J. Friedland, A survey of unmanned aerial vehicle (UAV) usage for imagery collection in disaster research and management, in: 9th International Workshop on Remote Sensing for Disaster Response, Vol. 8, 2011.
[35] F. Nex, F. Remondino, UAV for 3D mapping applications: a review, Appl. Geomat. 6 (1) (2014) 1–15.
[36] S. Chowdhury, W. Tong, A. Messac, J. Zhang, A mixed-discrete particle swarm optimization algorithm with explicit diversity-preservation, Struct. Multidiscip. Optim. 47 (3) (2013) 367–388.
[37] D. Mellinger, V. Kumar, Minimum snap trajectory generation and control for quadrotors, in: 2011 IEEE International Conference on Robotics and Automation, IEEE, 2011, pp. 2520–2525.
[38] C. Richter, A. Bry, N. Roy, Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments, in: Robotics Research, Springer, 2016, pp. 649–666.
[39] T. Luukkonen, Modelling and Control of Quadcopter, in: Independent Research Project in Applied Mathematics, vol. 22, Espoo, 2011.
[40] S. Chowdhury, V. Maldonado, R. Patel, Conceptual design of a multi-ability reconfigurable unmanned aerial vehicle (UAV) through a synergy of 3D CAD and modular platform planning, in: 15th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, 2014, p. 2178.
[41] A. Mehmani, S. Chowdhury, C. Meinrenken, A. Messac, Concurrent surrogate model selection (COSMOS): optimizing model type, kernel function, and hyper-parameters, Struct. Multidiscip. Optim. 57 (3) (2018) 1093–1114.
[42] A. Mehmani, S. Chowdhury, A. Messac, Predictive quantification of surrogate model fidelity based on modal variations with sample density, Struct. Multidiscip. Optim. 52 (2) (2015) 353–373.
[43] S.N. Lophaven, H.B. Nielsen, J. Søndergaard, et al., DACE: A Matlab Kriging Toolbox, Vol. 2, Citeseer, 2002.
[44] J. Van Den Berg, S.J. Guy, M. Lin, D. Manocha, Reciprocal n-body collision avoidance, in: Robotics Research, Springer, 2011, pp. 3–19.
[45] D. Bareiss, J. van den Berg, Generalized reciprocal collision avoidance, Int. J. Robot. Res. 34 (12) (2015) 1501–1514.



Amir Behjat is a PhD candidate in Mechanical and Aerospace Engineering at University at Buffalo. He graduated with his MS in Mechanical Engineering from Sharif University of Technology. He is currently working in the ADAMS Lab, under the supervision of Dr. Souma Chowdhury. His research interests include the following areas: Neuroevolution and Physics-cognizant machine learning, with applications in unmanned aerial systems and autonomy. He has co-authored one journal and seven full-length conference articles. He is a student member of the ASME and AIAA Societies.



Steve Paul graduated with a M.S. in Mechanical Engineering from University at Buffalo. His research interests included the following areas: network optimization and collision avoidance in unmanned aerial vehicles. He has co-authored one full-length conference article and two journal articles. His contributions to this article was made during his M.S. Thesis research on unmanned aerial vehicles.

Souma Chowdhury is an Assistant Professor of Mechanical and Aerospace Engineering at University at Buffalo, where he leads the Adaptive Design Algorithms, Models and Systems (ADAMS) Lab. He got his Ph.D. in Mechanical Engineering from Rensselaer Polytechnic Institute in 2012. His research is focused on evolutionary, neural, and swarm-intelligence algorithms for embodied artificial intelligence and design optimization. Primary areas of applications of his research include: i) design and autonomy of unmanned aerial vehicles or UAVs, ii) swarm robotics for environment monitoring and disaster response applications, iii) physics cognizant AI for dynamic and robotic systems. He has co-authored 31 international journal articles and 83 peer-reviewed conference articles in related topics. His research has been sponsored by NSF and DARPA. He is a member of the ASME, AIAA, and IEEE societies, and is responsible for organizing and chairing Data-driven Design, Multifidelity Modeling, and Artificial Intelligence sessions and workshops in various Society Conferences.