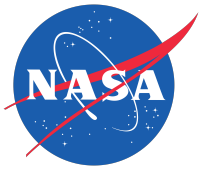
NASA/TM–20250009417



Foundational Simulation Modules for Future CMSA Safety Benefit Assessments

José Ignacio de Alvear Cárdenas
San José State University, Moffett Field, California

Seungman Lee
NASA Ames Research Center, Moffett Field, California

Priyank Pradeep
Analytical Mechanics Associates Inc, Moffett Field, California

Vincent H. Kuo
Metis Technology Solutions, Moffett Field, California

September 2025

NASA STI Program … in Profile

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA scientific and technical information (STI) program plays a key part in helping NASA maintain this important role.

The NASA STI Program operates under the auspices of the Agency Chief Information Officer. It collects, organizes, provides for archiving, and disseminates NASA's STI. The NASA STI Program provides access to the NASA Aeronautics and Space Database and its public interface, the NASA Technical Report Server, thus providing one of the largest collections of aeronautical and space science STI in the world. Results are published in both non-NASA channels and by NASA in the NASA STI Report Series, which includes the following report types:

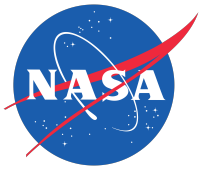
- TECHNICAL PUBLICATION. Reports of completed research or a major significant phase of research that present the results of NASA programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA counterpart of peer-reviewed formal professional papers, but having less stringent limitations on manuscript length and extent of graphic presentations.
- TECHNICAL MEMORANDUM. Scientific and technical findings that are preliminary or of specialized interest, e.g., quick release reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.
- CONTRACTOR REPORT. Scientific and technical findings by NASA-sponsored contractors and grantees.
- CONFERENCE PUBLICATION. Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or co-sponsored by NASA.
- SPECIAL PUBLICATION. Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.
- TECHNICAL TRANSLATION. English-language translations of foreign scientific and technical material pertinent to NASA's mission.

Specialized services also include creating custom thesauri, building customized databases, and organizing and publishing research results.

For more information about the NASA STI Program, see the following:

- Access the NASA STI program home page at *http://www.sti.nasa.gov*
- E-mail your question to *help@sti.nasa.gov*
- Fax your question to the NASA STI Information Desk at 443-757-5803
- Phone the NASA STI Information Desk at 443-757-5802
- Write to:
  STI Information Desk
  NASA Center for AeroSpace Information
  7115 Standard Drive
  Hanover, MD 21076–1320

NASA/TM–20250009417



Foundational Simulation Modules for Future CMSA Safety Benefit Assessments

José Ignacio de Alvear Cárdenas
San José State University, Moffett Field, California

Seungman Lee
NASA Ames Research Center, Moffett Field, California

Priyank Pradeep
Analytical Mechanics Associates Inc, Moffett Field, California

Vincent H. Kuo
Metis Technology Solutions, Moffett Field, California

National Aeronautics and Space Administration (NASA)

Ames Research Center, Moffett Field, California 94035

September 2025

Acknowledgments

The material is based upon work supported in part by NASA under award number 80NSSC22M0060 for San José State University Research Foundation.

Available from:

NASA Center for AeroSpace Information
7115 Standard Drive
Hanover, MD 21076-1320

National Technical Information Service
5301 Shawnee Road
Alexandria, VA 22312

Available electronically at http://www.sti.nasa.gov

Table of Contents

List of Figures

List of Tables

Acronyms and Definitions

ASTM American Society for Testing and Materials, a standards organization

BVLOS Beyond Visual Line of Sight

CBS Conflict-Based Search

CMSA Conformance Monitoring for Situational Awareness

CS Constraint Set

CT Constraint Tree

DF Duration-Fixed

FAA Federal Aviation Administration

GPS Global Positioning System

HNS Hub-and-Spoke

IMU Inertial Measurement Unit

LF Length-Fixed

MAC Mid-Air Collision

MBB Minimum Bounding Box

MILP Mixed-Integer Linear Programming

NAS National Airspace System

NASA National Aeronautics and Space Administration

OI Operational Intent

ONB Out-and-Back

OTD On-demand Time of Departure

OVB Operational Volume Block

P2P Point-to-Point

PDSTD Pre-Departure Strategic Temporal Deconfliction

SD Strategic Deconfliction

SLSQP Sequential Least Squares Programming

TCL Technical Capability Level

TLS Target Level of Safety

UAS Unmanned Aircraft System

USS UAS Service Supplier

UTM UAS Traffic Management

Foundational Simulation Modules for Future CMSA Safety Benefit Assessments

José Ignacio de Alvear Cárdenas[1], Seungman Lee[2], Priyank Pradeep[3], Vincent H. Kuo[4]

Summary

This work presents six foundational fast-time simulation modules for the future safety benefit study of UAS Traffic Management (UTM) Services, such as Conformance Monitoring for Situational Awareness (CMSA). First, a methodology is proposed to translate simulated or real data into Operational Volume Block (OVB) dimensions that comply with the ASTM conformance requirement of 95%. Fixing OVB blocks in length or duration led to the emergence of temporal and spatial overlap which are associated with contiguous and overlapping OVBs, respectively.

Second, realistic scenarios are automatically created by randomly and periodically selecting geographic locations (e.g. San Francisco or Houston) and spawning services that follow operational profiles that have been linked to actual geographic features. Here, we implemented hub-and-spoke, out-and-back and point-to-point operational profiles that can be connected to package delivery, inspection, and emergency medical supply transport services, respectively. Those services translate into flight paths that populate the airspace. Consequently, now practically an infinite number of scenarios can be simulated and it is possible to run Monte Carlo simulations whose conclusions can be generalized to the complete National Airspace System and will not potentially overfit to a few manually designed airspace geometries.

Third, it is shown how the dimensions of the OVB sizing stage are used to discretize the flight paths of the scenario generation, even around waypoints where the direction of flight changes.

Fourth, a low-latency 4D volume-based conflict detection module is proposed that uses R-trees. Three different implementations are considered. Results show that R-trees created with the OVBs of all flights in the airspace consistently lead to the lowest computational time when detecting conflicts, both during the initial airspace configuration and when incrementally adding new flights.

Fifth, a modified Conflict Based Search (CBS) approach is suggested for pre-departure 4D volume-based strategic temporal deconfliction. This method leverages R-tree-based conflict detection at the low level and employs a Constraint Tree structure at the high level. The cost function integrates both total airspace delay and the number of conflicts, with a higher weighting placed on conflict reduction, as achieving a conflict-free solution is the primary objective. Results show that it is able to resolve all conflicts in randomly generated scenarios, even when flight trajectories are diverse and unpredictable, including crossing, merging, and parallel flight paths. Moreover, it successfully deconflicts complex flight plans introduced into an already conflict-free airspace within seconds. Parallelizing the CBS algorithm introduces a beam search-like effect, which contributes to lower overall airspace delays by exploring multiple solution paths concurrently. However, this

[1]San José State University, Moffett Field, California
[2]NASA Ames Research Center, Moffett Field, California.
[3]Analytical Mechanics Associates Inc, Moffett Field, California
[4]Metis Technology Solutions, Moffett Field, California

benefit comes with a trade-off: the computational cost increases compared to the standard sequential greedy strategy.

Sixth, a scenario exporter module that allows researchers to import deconflicted scenarios in JSON format for use in any fast-time simulator that supports the ASTM UTM Protocol for OVB definition. The ultimate goal is to enable any simulator to import realistic, high-fidelity scenarios, allowing researchers to focus on the development and analysis of current and more advanced UTM functions, such as CMSA, instead of repeatedly building some of these supporting tools from scratch.

1 Introduction

Unmanned Aircraft Systems (UASs) are expected to generate substantial positive socio-economic impacts, which are projected to grow in the coming years (Ref. 2, 3). These impacts span both public applications like disaster management (Ref. 4), wildfire management (Ref. 5–7) and public safety (Ref. 8, 9), as well as commercial uses such as infrastructure inspection (Ref. 10, 11), agriculture (Ref. 12) and the delivery of package and medical supplies (Ref. 13, 14). According to the Federal Aviation Administration (FAA) Aerospace Forecast Fiscal Years 2025-2045 (Ref. 3), the United States is predicted to have 1.93 million recreational small drones and 1.18 million commercial drones by 2029.

To enable this routine, safe and efficient UAS operations, NASA launched in 2024 the UAS Traffic Management (UTM) Beyond Visual Line of Sight (BVLOS) project in collaboration with the FAA. UTM is an ecosystem envisioned as a highly automated, collaborative and distributed system designed to safely manage UAS operations in low altitudes minimizing direct human air traffic control (Ref. 15). This concept was primarily conceived and developed by NASA back in 2013 followed by Technical Capability Level (TCL) demonstrations that concluded in 2019 (Ref. 16–18). UTM BVLOS is an extension from that early work that aims at the operationalization of the concept, in part through its deployment with real commercial and public safety flights and services in the UTM Key Site Operational Evaluation in the Dallas-Fort Worth area (Ref. 19, 20). This evaluation aims to gather data to inform FAA rulemaking for BVLOS operations, becoming a template for national UTM deployment.

Besides FAA and NASA, industry partners have collaborated with these government agencies to develop the ASTM F3548-21 (Ref. 21) industry consensus standard that outlines the essential services that UAS Service Suppliers (USSs) must provide and how they interact within the UTM ecosystem. Conformance Monitoring for Situational Awareness (CMSA) is one of those services and it provides situational awareness for nonconforming or contingent UAS operations; states in which the UAS is predicted to or has deviated from its original authorized flight plan. Unfortunately, industry stakeholders lack consensus on the sufficiency of evidence in literature that demonstrates the added safety benefit of CMSA, which raises questions about warranting their required economical investment and added complexity to the UTM ecosystem.

This lack of consensus is exemplified by differing assessments of CMSA's incremental safety contribution. For instance, Appendix X4 of the ASTM standard (Ref. 21) develops a probabilistic analysis using a volume-based collision risk model to demonstrate the safety benefits of strategic deconfliction (SD), assuming compliance with the standard's provisions and certain operational parameters (e.g., off-nominal event rate and participation rate). The results show that SD alone can reduce the number of collisions by 97.9%, implying CMSA would only have the potential of improving safety by 2.1%. Similarly, Airbus (Ref. 22), using a simulation-based approach, concluded that SD alone could achieve a 99% reduction in Mid-Air Collisions (MAC), suggesting CMSA would potentially offer only an additional 1% safety benefit.

In contrast, NASA (Ref. 23) presented an analytical approach showcasing the need for additional deconfliction services beyond SD, such as CMSA, as a function of population density, off-nominal probability, and the number of flights per hour, in order to meet a Target Level of Safety (TLS). The results indicate a highly demanding off-nominal flight probability lower than 10^{-5} to meet the required TLS when the operational tempo is higher than 100 flights per hour in urban environments. Beyond their differing assumptions, the models also diverge in their methodology; the former

methods assess safety relative to a baseline, whereas the latter defines the requirement to meet an absolute TLS.

These analytical methods can be verified in simulation by directly computing the actual safety benefit contribution of CMSA. However, the main challenge is that its impact cannot be evaluated in isolation, as this service requires the integration of other UTM services. CMSA's function is limited to detecting UAS flight path deviations and generating predicted off-nominal Operational Intent (OI). That information needs to be ingested by other deconfliction services, such as pre-departure strategic deconfliction, to affect the 4D flight separation. Consequently, CMSA cannot be simulated in isolation and requires additional work beyond its direct implementation to derive conclusions.

Additionally, previous UTM simulation efforts evaluating safety have exhibited several constraints. These simulations have often been confined to specific airspace geometries and geographical locations (Ref. 22), lacked accounting for mission or service type diversity (Ref. 24), and utilized contiguous, homogeneous Operational Volume Blocks (OVBs) without correlation to actual vehicle performance (Ref. 25). Furthermore, these efforts have not adequately considered the heterogeneous landscape of UAS in the future National Airspace System (NAS) or the need for spatial and/or temporal OVB overlap (Ref. 26). As a result, the conflict detection and deconfliction services selected in these simulators tend to overfit scenario assumptions, rendering the derived CMSA safety benefits non-extrapolatable to the general airspace.

This research develops an initial set of simulation modules designed to overcome the limitations of previous efforts, serving as a critical first step towards higher-fidelity Monte Carlo simulations for comprehensively assessing the CMSA's safety benefits within the NAS. Overfitting to specific geographies or geometries is mitigated by enabling simulations across any major city, where diverse drone services (e.g., package delivery, inspection) are randomly generated and realistically linked to actual geographic features (e.g., supermarkets, restaurants, apartments). Moreover, an OVB sizing approach is proposed based on simulated or real flight data, directly reflecting vehicle performance. Finally, 4D volume-based conflict detection and pre-departure strategic temporal deconfliction (PDSTD) modules leveraging the R-Trees (Ref. 27) and Conflict-Based Search (CBS) (Ref. 28) heuristic algorithms were implemented. They can operate across arbitrary airspace geometries and are optimized for the high-speed performance typically needed in fast-time simulations.

The remainder of this paper is organized as follows. Section 2 outlines the methodology, elaborating on each of the simulation modules that shape the contribution of this work. Next, Section 3 presents the module outputs, visualized over the city of Mountain View, California, from flight path generation to strategically deconflicted OIs. Finally, Section 4 discusses the work's conclusions and provides recommendations for future research.

2 Methodology

Figure 1 shows all the blocks required to simulate a scenario and assess the safety benefit of CMSA. Given some user input and map data from OpenStreetMap, a scenario is generated in the chosen geographic location spawning flight paths from selected UAS mission profiles, such as hub-and-spoke (package delivery) or point-to-point (medical supply delivery). In order to discretize the paths, the OVBs corresponding to the performance of the chosen vehicle(s) are sized using simulated or flight data. Next, the constructed OIs are passed on to 4D volume-based conflict detection, for the discovery of all the intersections between OVBs of different flights, followed by 4D volume-

based pre-departure strategic temporal deconfliction where the departure times are adjusted. The conflict-free scenario is fed to the fast-time simulator (e.g., Fe^3 (Ref. 29)) which only needs to be paused when an off-nominal flight is injected. This would trigger the CMSA module which would detect the off-nominal behavior, transition the state of the flight to nonconforming or contingent, and generate off-nominal OVBs. The latter is fed back to the 4D volume-based PDSTD module before feeding the rescheduled remaining flights to the simulator, which will continue until the next failure is injected.

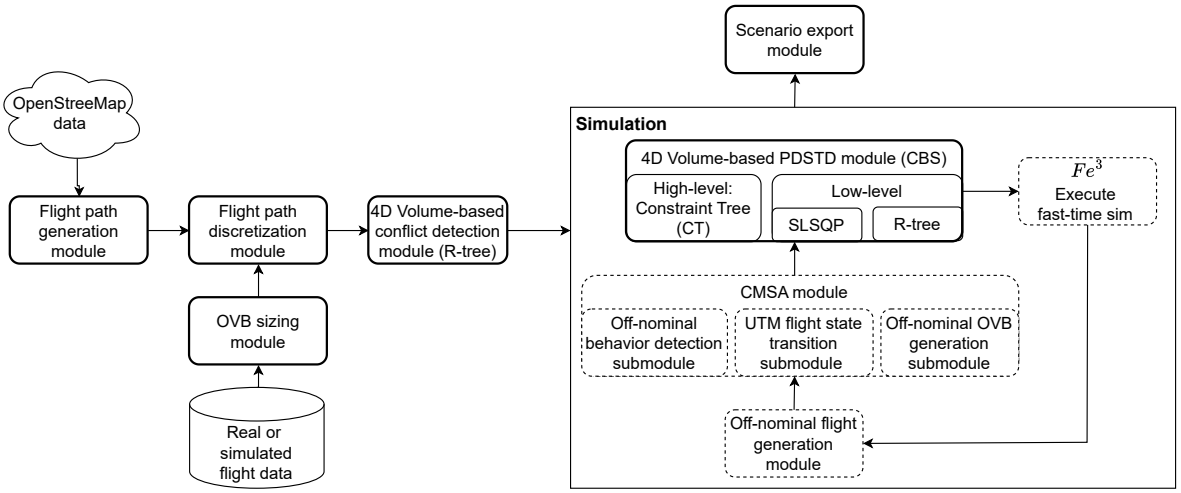


Figure 1. *CMSA simulation block diagram.*

The solid, rounded-edged blocks are the ones discussed in this section and their output will be visualized later in Section 3. The dashed blocks reflect the intention of future work.

2.1 OVB Sizing

The ASTM F3548-21 (Ref. 21) specification on UAS Traffic Management states that during the Activated state, a UAS should remain within the defined Operational Intent (OI) for at least 95% of its total flight time; this minimum compliance threshold is referred to as OiMinConformance in ASTM requirement OPIN0010. According to the UTM Concept of Operations (ConOps) (Ref. 15), the OI is composed of blocks that the research community understood as spatio-temporal 4D cuboids (Ref. 30–33) and called Operational Volume Blocks (OVBs) (Ref. 25). The standard does not prescribe a methodology for achieving this requirement, and the literature lacks a clear approach for determining the appropriate sizing of OVBs based on vehicle performance characteristics and system uncertainties. In most cases, the dimensions of OVBs are either assumed to be fixed and based on conservative estimates using factors such as the vehicle's cruise speed (Ref. 25, 34), or determined through a grid search over a range of UAS noise values and OVB dimensions (Ref. 22).

To address this gap in the literature, a simple methodology is proposed using the error between the desired and actual flight trajectories of the vehicle, which can be obtained in simulation or from real flights. For illustration purposes, cruise trajectories are simulated using the simple kinematic model shown in Equation 1. The pitch (θ), heading (ψ) and velocity (V) are sampled at every time step from a Gaussian distribution with the mean and standard deviations shown in Table 1.

$$\boldsymbol{X_{t+1}} = \boldsymbol{X_t} + \begin{bmatrix} |V_t| \cdot \cos\theta_t \cdot \cos\psi_t \\ |V_t| \cdot \cos\theta_t \cdot \sin\psi_t \\ |V_t| \cdot \sin\theta_t \end{bmatrix} \cdot dt \tag{1}$$

Table 1. *Kinematic model inputs*

	Velocity [m/s]	**Heading [deg]**	**Pitch [deg]**
Mean | Std	10 | 2	0 | 10	0 | 2

Figure 2 shows 100 trajectories projected onto the x-y plane, with each point also associated with a time component. In real-world scenarios, position drift caused by external factors such as wind, or by internal factors such as sensor inaccuracies (e.g. inertial measurement unit or IMU drift), is typically compensated by the onboard controller and complementary sensors (e.g., GPS). In this simple simulation, drift in the y and z axes is corrected by clipping each trajectory to the centerline every time it enters a new OVB, as will be seen in Section 3.1.

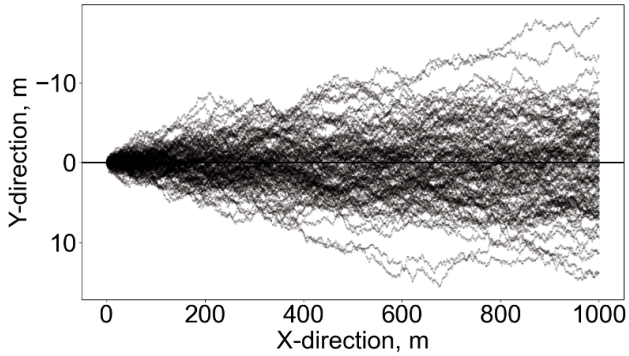


Figure 2. *Simulated trajectories without drift correction.*

Table 2. *OVB sizing methods by input and the type of emerging OVB overlap*

Sizing input	**OVB overlap type**
Length	Temporal
Duration	Spatial

OVBs can be sized by fixing their length (meters) or duration (seconds). In the case that the length is fixed, the trajectories are chopped along the x-axis and the data points within each segment are used to determine each OVB's dimensions. Besides the 95% conformance rate required by the standard (Ref. 21), it defines the OI as a "*volume-based representation of the intent for a*

UAS operation; comprises one or more overlapping or contiguous 4D volumes, where the start time for each volume is the earliest entry time, and the stop time for each volume is the latest exit time". Hence, the entry and exit times of each OVB correspond to the earliest and latest time steps of the data points within each segment. Since the x-dimension is fixed, the y and z dimensions are determined so that 95% of the data points lie within the OVB. To approximate this joint coverage, each marginal (y and z) dimension is bounded using a symmetric interval around the mean that contains $\sqrt{95} = 97.5\%$ of the data points in that dimension. This empirical method yields a rectangular approximation of the joint distribution by independently computing tolerance intervals along each axis. To further refine these last two dimensions, a global optimization approach such as Particle Swarm Optimization (PSO) (Ref. 35, 36) can be applied. The objective function minimizes the difference between the actual proportion of points contained within the OVB and the target value of 95%. The empirically computed bounds from the previous step serve as initial estimates for the particle positions in PSO.

When the OVB length is fixed and trajectories are clipped at the OVB exit, each trajectory will have a different exit time. As a result, and following the standard definition of the OI, consecutive OVBs exhibit **temporal overlap** (Table 2). This emerging overlap arises because the exit time of one OVB may extend beyond the entry time of the next.

In the case that the OVB duration is fixed, trajectories are sliced along the temporal axis, and only the data points time stamped within the OVB's entry and exit times are used for its sizing. The method is identical to length-fixed sizing, except that the dimension along the x-axis is not predefined. Following the standard's OI definition, the length of each OVB is determined such that it fully encloses all its corresponding trajectory points. Finally, since trajectories are sliced at different locations along the x-axis due to the fixed duration, a **spatial overlap** (Table 2) emerges between OVBs, as the exit location of one may extend beyond the entry location of the next.

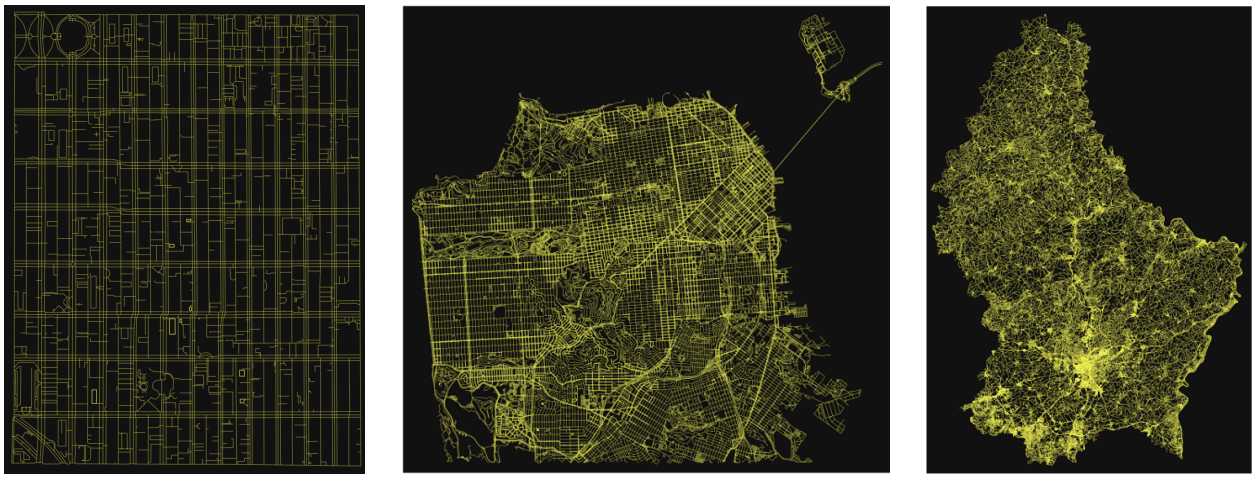
The output of the OVB sizing are six variables, namely the length (l), width (w), height (h) and duration (T) of the block, as well as its spatial (O_s) and temporal (O_t) overlaps.

2.2 Flight Path Generation

Current simulation approaches use a manually generated geometry for the generation of their airspace traffic (Ref. 24–26) or are constrained to a particular geography (Ref. 22), hindering the extrapolation of the conclusions to the complete NAS. The results overfit the fine-tuned complexity of the designed scenarios (e.g., airspace density, number and type of crossing points) and assumptions (e.g., only straight flights considered) which sometimes can be unrealistic.

To address these challenges and assess the safety benefits of CMSA our scenario generation incorporates two distinct points of randomization. The first point of randomization pertains to the geographical location itself. To assess the safety benefit of CMSA agnostic to any single geography, our Monte Carlo simulation can change the area of operation after a certain number of iterations. For instance, in the context of the NAS, this would mean alternating between all or major cities of the USA. The user can define these specific locations that span from neighborhoods to cities (e.g., "Capitol Hill, Denver, USA," "San Francisco, California, USA," or "Japan"), which are then used by the OSMnx library (Ref. 37) to import and spawn the corresponding geographical area, as seen in Figure 3.

The second point of randomization involves the stochastic spawning of flight paths within the chosen airspace, following predefined operational profiles linked to services. Once a geographical area is established through the first randomization step, the flight paths of vehicles are randomly



(a) Capitol Hill, Denver, USA *(b) San Francisco, California, USA* *(c) Luxembourg*

Figure 3. *Grid plans from OSMnx.*

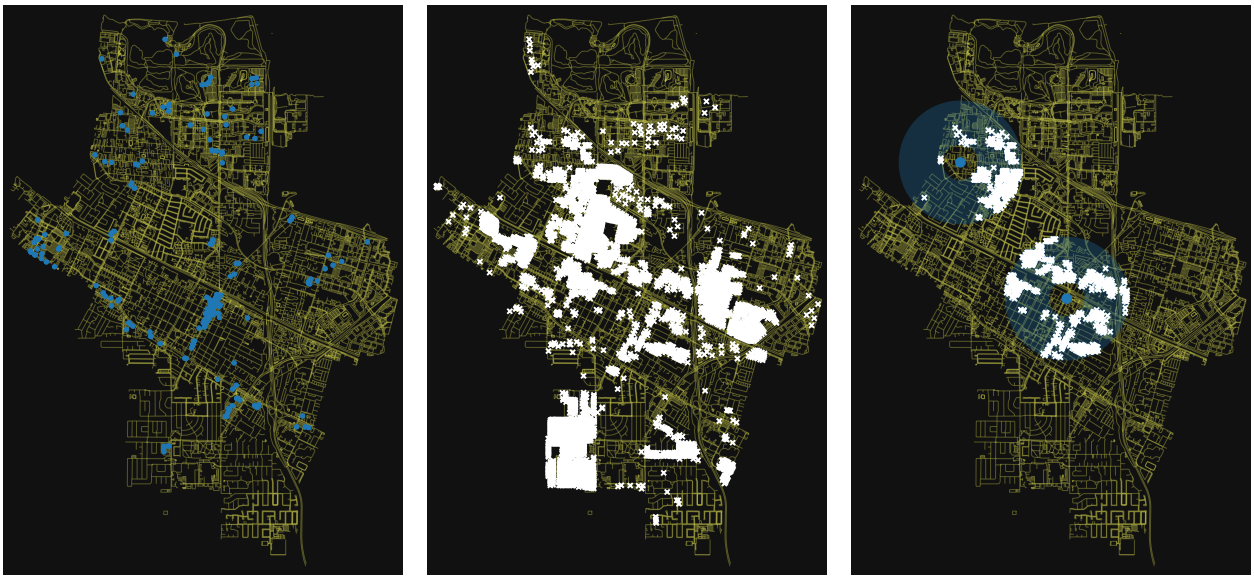
generated within that specific region. These services are specified to sometimes leverage information from the ground, such as the location of warehouses or apartment buildings. This ensures a realistic, diverse and unpredictable set of flight trajectories for robust scenario testing, relevant to the characteristics of the selected location.

For the current version of this simulation module, three operational profiles are considered, namely hub-and-spoke (HNS), out-and-back (ONB) and point-to-point (P2P). Their On-demand Time of Departure (OTD) is simulated separately using a Poisson distribution (Ref. 25, 34, 38). Each will be briefly discussed in the following subsections.

2.2.1 Hub-and-Spoke

HNS is typically associated with package delivery. It consists of a central departure location called depot where multiple UAS take-off towards drop-off locations (clients) within a certain range. A Company operating HNS can have multiple depots separated by a minimum distance from each other. Leveraging the real-world information from the geographical location through OSMnx, depots are located at the coordinates of restaurants, bars, cafes, pubs, supermarkets or shops, or a subset of those. Meanwhile, a potential client can be a residential building, hotel, office, college, university, stadium, dormitory, house, bungalow or static caravan within the radius of operation of the depot. This region of operation is shaped as a ring, allowing for a minimum and maximum distance from the depot. Figure 4a and Figure 4b show the locations of potential companies and clients in the area of Mountain View, California, whereas Figure 4c shows the potential clients within the region of operation of each depot. Depending on emerging services, the physical meaning of depots and clients can be changed in the future by using different OpenStreetMap tags.

In Table 3 are the inputs for the generation of HNS missions. For every scenario, a user-defined number of depots per company is chosen at random from the potential candidates in the region. Only when a depot has enough potential clients within its operational range, it is approved for flight generation and client locations are selected at random as flight targets.



(a) *Potential depots (blue circles)* ***(b)*** *All clients (white crosses)* ***(c)*** *Depot potential clients*

Figure 4. *Scenario generation in Mountain View.*

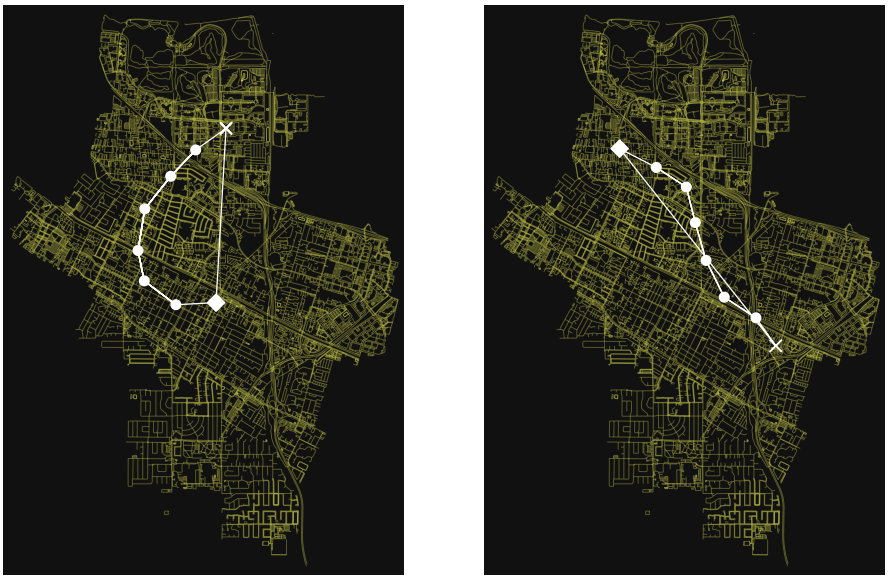
Table 3. *Hub-and-spoke inputs*

Number of companies	Depot tags	Max. radius of operations
Number of depots per company	Client tags	Min. radius of operations
Number of flights per depot	Intra-company depot distance	Average take-off rate (Poisson distribution)
OVB dimensions per depot	Min. clients per depot	Depot min. temporal flight separation

2.2.2 Out-and-Back

ONB operations can be associated, among others, with the inspection of an electrical grid, perimeter patrol or real estate videography. As can be seen in Figure 5, it consists of a flight path that starts at a location (diamond), flies following a certain number of waypoints (circles) where heading changes can be performed and concludes (at the white cross) returning in a straight line to the point of origin for landing. The user can specify the number of waypoints, the spacing between them, the maximum allowed heading change at each waypoint, and whether the vehicle must turn in a single direction or is allowed to zigzag.

Table 4 lists the inputs for the generation of ONB missions. Multiple companies can operate several ONB flights. For the current implementation, ONB flights have their departure/landing and waypoints coordinates randomly chosen on land within the region of interest (e.g., Mountain View), with waypoint selection stopping after a set number of failed attempts. However, instead of using random sampling, the structured, topological and attribute-rich spatial data from OpenStreetMap



(a) *Turning in a single direction* ***(b)*** *Zigzagging*

Figure 5. *Out-and-back flight profile examples.*

(via OSMnx) could be leveraged to define rules that enable more realistic simulation of one or more of the aforementioned services. For example, the tags "power=line", "landuse=residential" and "building=villa" could guide UAS to follow power lines or orbit residential areas or villas.

Table 4. *Out-and-back inputs*

Number of companies	Min. number of waypoints	Waypoint search limit
Number of flights per company	Max. inter-waypoint distance	Turn single direction flag
OVB dimensions per company	Min. inter-waypoint distance	Average take-off rate (Poisson distribution)
Max. number of waypoints	Waypoint max. heading change	

2.2.3 Point-to-Point

P2P operations are linked to services such as emergency medical supply transport between healthcare facilities, transport of lab samples to diagnostic centers or disaster relief supply drop-offs. It is the simplest operational profile from the ones implemented in this work. As can be observed in Figure 6, it consists of a straight flight path from a take-off point (diamond) to a landing point (cross). The distance between these locations is user-configurable.

In Table 5 are the inputs for the generation of P2P missions. Again, for the work shown here, the two points of interest are chosen at random on land within the region of interested. When searching for points that meet those conditions, the algorithm samples fewer than a specified number of candidates before giving up. To define rules for more realistic simulation of the aforementioned services, OpenStreetMap tags such as "amenity=hospital", "healthcare=laboratory" and "emergency=assembly_point" could be used.

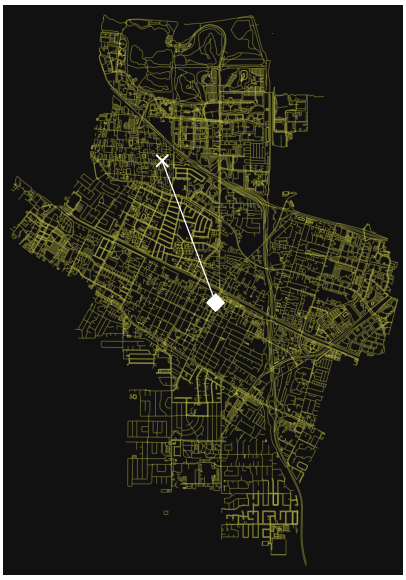


Figure 6. *Point-to-point flight profile example.*

Table 5. *Point-to-point inputs*

Number of companies	Max. flight distance	Average take-off rate (Poisson distribution)
Number of flights per company	Min. flight distance	
OVB dimensions per company	Take-off/Landing search limit	

2.3 Flight Path Discretization

Once the OVB dimensions are defined, flight path discretization is the problem of spatially positioning OVBs along a flight path, as well as defining their entry and exit times. Previous UTM efforts (Ref. 25, 38) assumed straight-line flight paths composed of contiguous OVBs, with each OVB placed tangentially to the one before it. Under this assumption, only the final block required spatial and temporal trimming to align with the remaining portion of the flight path. In this work, due to the addition of waypoints, there are four scenarios that will be discussed in the next sections.

2.3.1 Starting OVB

In this work, UAS are assumed to take-off vertically, and OVBs begin forming above the take-off location once the cruise altitude is reached. The distance from the start of cruise to the center of the first OVB ($d_{0,1}$) is defined as half the length of that OVB (l_1), as shown in Equation 2. The entry time of the first OVB corresponds to the start of the cruise phase (t_0), and the exit time is given by the entry time plus the duration of the first OVB (T_1), as determined during the sizing stage (Equation 3).

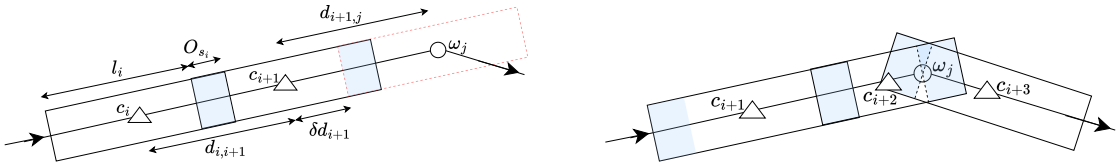
$$d_{0,1} = \frac{l_1}{2} \tag{2}$$

$$\begin{aligned} t_{0_1} &= t_0 \\ t_{e_1} &= t_{0_1} + T_1 \end{aligned} \tag{3}$$

2.3.2 OVBs Between Waypoints

The distance along the flight path between the centers of two OVBs ($d_{i,i+1}$) is computed in Equation 4, taking into consideration the spatial overlap (O_{s_i}). The variables l_i and l_{i+1} represent the length of two consecutive OVBs, and δd_i is the distance from the previous OVB to the start of the spatially overlapped region, as can be seen in Figure 7a.

$$\begin{aligned} \delta d_i &= \frac{l_i}{2} - O_{s_i} \\ d_{i,i+1} &= \delta d_i + \frac{l_{i+1}}{2} \end{aligned} \tag{4}$$



(a) *Between waypoints* ***(b)*** *Around waypoint*

Figure 7. *Flight path discretization.*

The OVBs must also be temporally aligned along the flight path. Equation 5 calculates the center time ($t_{c_{i+1}}$) of each OVB between waypoints, taking into account the temporal overlap (O_{t_i}). Let T_i and T_{i+1} represent the duration of the previous and current OVBs, respectively. It also computes the OVB's entry ($t_{0_{i+1}}$) and exit ($t_{e_{i+1}}$) times, with $t_{c_{i+1}}$ defined as their average.

$$\begin{aligned} t_{c_{i+1}} &= t_{c_i} + \frac{T_i + T_{i+1}}{2} - O_{t_i} \\ t_{0_{i+1}} &= t_{c_{i+1}} - \frac{T_{i+1}}{2} \\ t_{e_{i+1}} &= t_{c_{i+1}} + \frac{T_{i+1}}{2} \end{aligned} \tag{5}$$

2.3.3 OVB Before Waypoint

An OVB is placed just before a waypoint when the condition in Equation 6 is satisfied. Specifically, this occurs when the distance from the previous OVB center to the end of the current OVB — excluding the portion that overlaps with the potential next OVB — exceeds the available space between the previous OVB center and the upcoming waypoint ($d_{i,j}$).

$$d_{i,i+1} + \frac{l_{i+1}}{2} - O_{s_i} \leq d_{i,j} \tag{6}$$

When reaching a waypoint, the OVB is trimmed such that only the portion that would be under spatial overlap overshoots the waypoint, as can be observed in Figure 7b. In that case, the distance of that OVB center to the previous one ($d^*_{i,i+1}$) is computed using Equation 7. Here, l^*_{i+1} is the length of the trimmed OVB.

$$\begin{aligned} l^*_{i+1} &= d_{i,j} - \delta d_i + O_{s_i} \\ d^*_{i,i+1} &= \delta d_i + \frac{l^*_{i+1}}{2} \end{aligned} \tag{7}$$

Since the OVB is trimmed, the center ($t^*_{c_{i+1}}$), exit ($t^*_{e_{i+1}}$) and entry times ($t^*_{0_{i+1}}$) are computed using Equation 8. The quantity δt_i denotes the average time it should take the vehicle to travel from the previous OVB center to the start of the spatially overlapped region; l_{i+1} and T_{i+1} represent the length and duration of the OVB if it were not trimmed; and T^*_{i+1} denotes the duration of the trimmed OVB.

$$\begin{aligned} \delta t_i &= \tfrac{T_i}{2} - O_{t_i} \\ T^*_{i+1} &= l^*_{i+1} \cdot \tfrac{T_{i+1}}{l_{i+1}} \\ t^*_{c_{i+1}} &= t_{c_i} + \delta t_i + \tfrac{T^*_{i+1}}{2} \\ t^*_{0_{i+1}} &= t^*_{c_{i+1}} - \tfrac{T^*_{i+1}}{2} \\ t^*_{e_{i+1}} &= t^*_{c_{i+1}} + \tfrac{T^*_{i+1}}{2} \end{aligned} \tag{8}$$

2.3.4 OVB After Waypoint

The first OVB after the waypoint is not trimmed but it is shifted back by δd_i from the waypoint such that the spatial overlap is behind the waypoint. Finally, Equation 9 computes the center time of the first OVB after the waypoint by first computing the time that the vehicle is expected to fly over the waypoint (t_j). The entry and exit times are computed using the equations in Equation 5.

$$\begin{aligned} t_j &= t^*_{c_i} + (\tfrac{T^*_i}{2}) - O_{t_i} + O_{s_i} \cdot \tfrac{T_{i+1}}{l_{i+1}} \\ t_{c_{i+1}} &= t_j + \delta t_{i+1} \end{aligned} \tag{9}$$

2.3.5 Buffers

Once the OVBs have been positioned in space and time, it is possible to add buffers in all 4 dimensions. Additionally, the approach discussed in the previous sections enables the researcher to have both spatial and temporal overlaps even though in the sizing process only one emerges per chosen input. The buffers do not change the spatial position of the OVB centers but may increase the safety during deconfliction by ensuring an increased separation.

2.4 4D Volume-Based Conflict Detection

Per the ASTM F3548-21 standard (Ref. 21), a conflict occurs when the OVBs of two different flights overlap in 4D, intersecting in both space and time: "*the spatial dimensions of the 4D volumes must share at least one point and the start/end time range for the two 4D volumes must overlap*". Given that scenarios are automatically generated by spawning flights from diverse operational profiles, it is necessary to implement a conflict detection approach that does not rely on airspace geometry assumptions. Crossing points, parallel flights and merging lanes could emerge in a scenario. To address this challenge, the approach proposed here is based on R-trees (Ref. 27).

The R-tree is a hierarchical data structure specifically designed to index and query multi-dimensional information, making it highly effective for spatial data like geographic coordinates, polygons, or other geometric shapes. They have been commonly researched and used in (spatial) databases (Ref. 39, 40). At the core of its design is the Minimum Bounding Box (MBB) (Ref. 1),

which in 2D is the smallest possible rectangle that encloses a collection of nearby spatial objects, as can be seen in Figure 8.

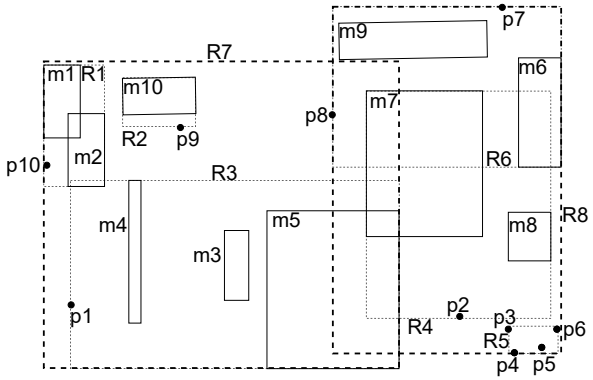


Figure 8. *Minimum Bounding Boxes (Ref. 1).*

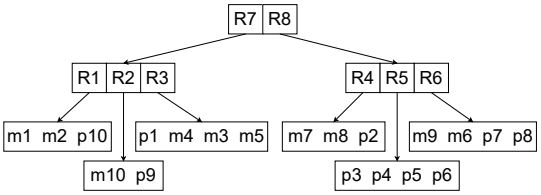


Figure 9. *R-tree data structure (Ref. 1).*

The R-tree indexes information in a dynamic, height-balanced tree (Ref. 27). It efficiently handles a combination of insertions, deletions, and searches without the need for regular structural adjustments, and the height difference between the left and right subtrees of any node is bounded, ensuring balanced performance across operations. Figure 9 shows how this structure is organized in levels (Ref. 1). At the lowest, leaf nodes contain the actual data objects (or pointers to them). Moving up the hierarchy, parent nodes do not store objects directly; instead, they store the MBBs that fully enclose the geographic regions of their child nodes. This nested structure is the key to its performance. When a spatial query is executed (e.g., finding all bike shops within a one-kilometer radius), the search algorithm can quickly discard large portions of the data. It starts at the root node and recursively descends the tree, checking if the query region overlaps the bounding boxes of the nodes. If the query area does not intersect with an MBB at a high-level node, the R-tree does not need to examine any of the branches beneath it, dramatically reducing search time.

The process of constructing an R-tree (Ref. 1, 27) begins with an empty root node that is also a leaf node and has no entries. Once the first data object (e.g., a point or polygon) is inserted, it is added to the root/leaf node. The node's MBB is set to be the same as the object's MBB.

For each subsequent object, the tree performs the following three-step insertion procedure. First, a subtree is chosen. Starting at the root, the algorithm selects the child node whose MBB requires the least amount of enlargement to enclose the new object. This process is repeated down the tree until a leaf node is reached. Second, the new object is added to the selected leaf node. Third, the MBB of the leaf node is adjusted (if necessary) to enclose the new object. This change is then propagated up the tree, and each parent MBB is updated to reflect the new size of its child's MBB.

Each node in an R-tree has a maximum capacity. If adding a new object to a leaf node causes it to exceed this capacity, a split occurs. The entries in the overflowing node (including the new object) are divided into two groups. The goal of the split algorithm (e.g., linear or quadratic (Ref. 27)) is to create two new nodes whose MBBs cover the minimum possible area and have the least overlap. The single MBB in the parent node is replaced with two MBBs representing the two new nodes. This can cause the parent node to overflow, triggering another split. This process can continue all the way up to the root. If the root node itself is split, a new root node is created with two children (the two nodes resulting from the split). This is how the R-tree increases in height.

In the present research, R-trees are employed for 4D volume-based conflict detection to identify overlapping OVBs from different flights in both space and time. The baseline method uses a naive approach that compares each OVB from each flight with all OVBs from every other flight. The process involves a sequential filtering: it first checks for temporal overlap, followed (if successful) by an altitude overlap check, and finally, if both conditions are met, it performs a 2D polygon intersection to confirm spatial conflict. Since that can be computationally expensive, three alternative R-tree conflict detection implementations are considered:

1. **Hierarchical OI/OVB R-trees**: a high-level R-tree is built by inserting each flight's OI as a single volume. In addition, a low-level tree is built for each flight containing its respective OVBs as entries. Each OVB from every flight is then used to query the high-level R-tree, which returns a list of flights that may be in conflict. Excluding its own flight, the same OVB is subsequently used to query the low-level trees of these candidate flights, yielding a refined list of potentially conflicting OVBs. Since the R-tree returns only those candidates whose MBB intersects the query OVB, a final naive sequential filtering step is then applied to each of these candidates to confirm actual OVB overlap.

2. **Hybrid OI R-tree/naive**: an R-tree is built as the high-level tree in the hierarchical implementation. However, instead of building and querying low-level trees, the query OVB is directly checked for conflicts against all OVBs of the candidate flights using the naive sequential filtering approach.

3. **OVB R-tree**: an R-tree is built with all the OVBs in the airspace, and each OVB from every flight is used to query the tree. To avoid self-comparison, OVBs belonging to the same flight are excluded from the list of intersection candidates. A final naive sequential filtering step is applied to each remaining candidate; the detailed check to confirm actual conflicts.

The R-trees of all these approaches have been implemented as 4D R-trees using the efficient *rtree* Python library[¶], a wrapper of the C++ libspatialindex library. This enables volume-based conflict detection across all dimensions with a single search query. The performance of these three approaches will be compared to the naive baseline later in Section 3.4.

Finally, the benefits of R-trees are efficiency, flexibility, and scalability. They promise faster spatial queries compared to flat spatial databases, such as the naive baseline approach. R-trees can handle various OVB shapes, including both trajectory-based and area-based representations. They are also well-suited for scenarios involving large numbers of flights, as their search time increases logarithmically ($\log_m N$, where m is the node capacity and N is the number of OI/OVBs), in contrast to the quadratic complexity of the naive approach ($O(N^2)$).

2.5 4D Volume-Based Pre-Departure Strategic Temporal Deconfliction

Previous UTM work in pre-departure strategic temporal deconfliction (PDSTD) used mixed-integer linear programming (MILP) approaches (Ref. 34), either alone or in combination with heuristics such as rolling horizon methods (Ref. 38), to determine vehicle departure times that ensure conflict-free airspace prior to takeoff. These methods are classified as temporal deconfliction strategies, as they adjust departure times instead of flight paths or speeds.

[¶]https://rtree.readthedocs.io

Classical MILP-based approaches that seek the global optimum exhibit poor scalability, as the formulation grows rapidly with the number of vehicles and spatial intersections, resulting in a large number of variables and constraints per vehicle. Hence, these methods suffer from constraint explosion, requiring all possible spatial interactions to be predefined, even if many conflicts never occur. They are also inflexible in dynamic environments, as MILP solvers are optimized for static problems and must be entirely reformulated and re-solved when new vehicles or conditions arise. Furthermore, they remain computationally inefficient in sparse conflict scenarios, processing the full optimization space regardless of actual conflict density. In addition to these computational and modeling drawbacks, existing MILP formulations in the UTM literature have not addressed the resolution of 4D volume-based conflicts arising from merging or parallel flight paths, but have mostly focused on 3D conflicts (2D in space) at crossing points.

This work proposes an alternative approach based on Conflict Based Search (CBS) (Ref. 28) that circumvents these limitations, albeit at the cost of sacrificing global optimality in favor of locally optimal solutions. Originating from the field of Multi-Agent Path Planning, CBS decomposes a large, hard problem into smaller, more tractable subproblems, resolving the conflicts emerging in the scenario incrementally. Instead of introducing all scenario constraints upfront, it addresses one conflict at a time, incorporating only the relevant constraints needed to resolve it. Unlike methods such as MILP, which grow directly with the number of agents and spatial intersections, CBS only scales with the number of conflicts. Even though heuristics can break a large MILP into smaller instances, effectively reducing the problem size and the computational time, it still requires defining all the airspace constraints upfront.

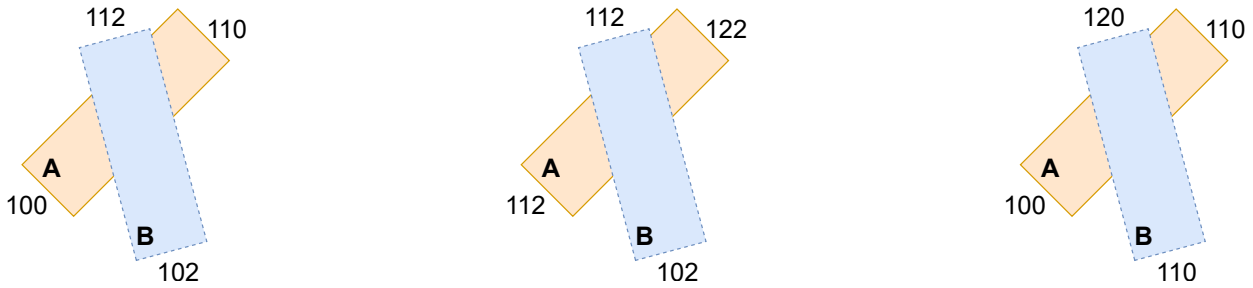
CBS enhances scalability (larger agent sets) by introducing constraints only when conflicts arise, resulting in a sparse, focused search space. It is also well-suited for dynamic environments, supporting easy integration of new aircraft in the airspace or local replanning. It is especially efficient in low-conflict scenarios, as it limits revisions to only a subset of flights.

The algorithm operates on two levels: a high-level search over a conflict tree that branches on 4D volume-based detected conflicts, and a low-level optimizer responsible for computing conflict-free OTDs satisfying individual vehicle constraints. The following sections detail each component.

2.5.1 High-Level: Conflict Tree Search

At the high level, CBS operates using a constraint tree (CT), where each node represents a set of constraints on agents' departure times. The tree is binary, and each node contains: (1) a set of constraints, (2) a solution specifying the departure times of all agents that satisfy those constraints, and (3) the total cost of the current solution.

The root contains an empty set of **constraints**. Each child node inherits the constraints of its parent and adds new constraints to resolve a single conflict. In this work, when a conflict occurs, as seen in Figure 10a, there are two possible resolutions: either vehicle A or vehicle B is delayed. If vehicle A is delayed (Figure 10b), the constraint set (CS) in Equation 10 is added to one of the two child nodes. Conversely, if vehicle B is delayed (Figure 10c), the CS in Equation 11 is added to the other. Here, STD_X stands for Scheduled Time of Departure for flight X, whereas X_0 and X_e are the entry and exit times, respectively, of the OVB of flight X involved in the conflict. Δ is a constraint constant computed once when resolving the conflict.



(a) Conflict *(b) A is delayed* *(c) B is delayed*

Figure 10. *Conflict between Flight A (orange OVB with solid outline) and Flight B (blue OVB with dashed outline), with their temporal resolution options. Values indicate entry and exit times.*

$$\begin{aligned} STD_A &> STD_B + \Delta \\ \Delta &= (STD_A - STD_B) - (B_e - A_0) \end{aligned} \tag{10}$$

$$\begin{aligned} STD_B &> STD_A + \Delta \\ \Delta &= (STD_B - STD_A) - (A_e - B_0) \end{aligned} \tag{11}$$

On top of those constraints, each flight has the added constraint that the STD has to be later than the OTD from Section 2.2: $STD_A > OTD_A$ and $STD_B > OTD_B$.

The **solution** (S) of a node is the list of vehicle STDs that satisfy its constraints, computed by the algorithm's low-level component. Its **cost** (C), defined in Equation 12, combines the total number of resulting conflicts (n_{conf}) and the aggregate delay across all vehicles (D). The delay of a vehicle is defined as the difference between its scheduled and on-demand times of departure ($D_X = STD_X - OTD_X$). Conflicts are assigned a higher penalty using a Big-M constant (M), reflecting the priority of achieving a conflict-free solution.

$$C = M \cdot n_{\text{conf}} + D \tag{12}$$

Besides the tree, a priority queue Q stores unexplored leaf nodes, ordered by cost from lowest to highest. This enforces a greedy strategy: nodes with the fewest conflicts are prioritized, and ties are broken by selecting the node with the lowest delay.

The high-level algorithm begins by creating a root node without constraints. Its solution consists of the vehicles' ODTs, and the cost is based solely on the number of conflicts detected with the efficient R-trees described in Section 2.4 (see Figure 1). Since vehicles depart at their ODTs, there is no delay. This root node is then added to Q.

The lowest-cost leaf node (initially the root) is dequeued from Q. If it has zero conflicts ($n_{\text{conf}} = 0$), the scenario is deconflicted, the high-level algorithm terminates, and its solution is returned. Otherwise, the earliest conflict is selected for resolution, and the two CSs, defined in Equation 10 and Equation 11, are computed. Two child nodes are spawned, each inheriting the parent's constraints plus one of the two CSs. The low-level algorithm is then executed for both children, and they are added to Q. This process repeats by dequeuing the next lowest-cost node from Q. Algorithm 1 shows the pseudo-code for the high-level algorithm.

Algorithm 1 High-Level Algorithm Pseudocode

Create initial solution with flight path generation (see Section 2.2): $S_0 = [OTD_1, OTD_2, \ldots]$
Compute the initial number of conflicts n_{conf_0} with R-trees (see Section 2.4)
Compute initial cost $C_0 = M \cdot n_{\text{conf}_0}$
Create root node N_0 with S_0, n_{conf_0} and C_0
Add N_0 to Q
Initialize best cost $C_{\text{best}} \leftarrow \infty$
Initialize best solution $S_{\text{best}} \leftarrow \emptyset$
while Q is not empty **do**
  Extract node N_X with lowest cost C_X from Q
  if $n_{\text{conf}_X} = 0$ **then**
    return Solution S_X
  else if $C_X < C_{\text{best}}$ **then**
    $C_{\text{best}} \leftarrow C_X$
    $S_{\text{best}} \leftarrow S_X$
  end if
  Choose earliest conflict in N_X
  Compute the two CSs to resolve the conflict (see Equation 10 and Equation 11)
  Spawn 2 child nodes, each with one CS
  for each child node N_i **do**
    Run low-level algorithm to compute solution and cost for N_i (see Section 2.5.2)
  end for
  Add child nodes with solution to Q
end while
return Best stored solution S_{best}

2.5.2 Low-Level: Individual Conflict Resolution

Given a child node with its constraints from the high-level algorithm, the low-level algorithm computes a solution that satisfies these constraints and evaluates its cost. Only the flight affected by the last added constraint is updated, as it was involved in the conflict of the parent node.

To resolve the scheduling constraints for the most recently constrained vehicle pair, a nonlinear optimization problem is formulated and solved using Sequential Least Squares Programming (SLSQP). The goal is to minimize D_X of the delayed vehicle X by changing the decision variable STD_X, subject to its node-specific constraints. If no feasible solution is found, the node is discarded. Otherwise, the vehicle's OI and OVBs entry and exit times are updated in the R-tree, which in turn updates the set of conflicts caused by that vehicle. These updates are then used to calculate the conflict count and delay cost. Finally, the node is added by the high-level algorithm to Q for further consideration as a potential parent node.

2.6 Scenario Exporter

Once the scenario has been generated and, if needed, deconflicted, it can be passed to a fast-time simulator such as Fe^3 (Ref. 29) to model vehicle dynamics and environmental effects like wind (Ref. 26). To support this, the OVBs for all flights, regardless of service type or operational

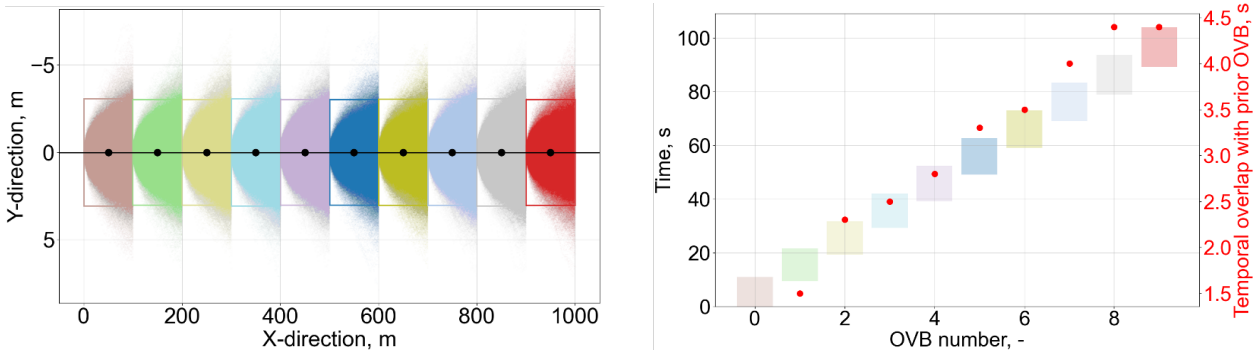
profile, are exported in JSON format, following the ASTM UTM Protocol[‖]. This enables any simulator to import realistic, high-fidelity scenarios, allowing researchers to focus on their core research questions rather than repeatedly building supporting tools from scratch. Without this capability, researchers are often forced to re-implement the same foundational simulation modules with limited time and resources, resulting in ad-hoc, low-fidelity solutions that compromise the overall quality and reproducibility of the work.

3 Results: Module Outputs and Design Justifications

This section presents an example that demonstrates the output of each module and explains the rationale behind certain design choices, such as the chosen final R-tree implementation. All tests were run on a macOS machine with a 2.7 GHz Quad-Core Intel Core i7 processor (4 cores, 8 logical processors), 16 GB of LPDDR3 RAM, running macOS Sequoia 15.6.

3.1 OVB Sizing

When the OVB length is fixed at 100 meters and the kinematic model is initialized using the inputs from Table 1, the resulting 10 OVBs — generated by sampling 10,000 trajectories of 1,000 meters — are shown spatially and temporally in Figure 11. As mentioned in Section 2.1, trajectories are clipped to the centerline at each OVB entrance. The OVBs' length remains constant, whereas their width and height vary around a mean. OVB duration and temporal overlap increase linearly due to the uncorrected longitudinal drift, an observation expected to vanish or dampen in higher-fidelity simulations or real-world scenarios where drift is corrected by the vehicle's controller.



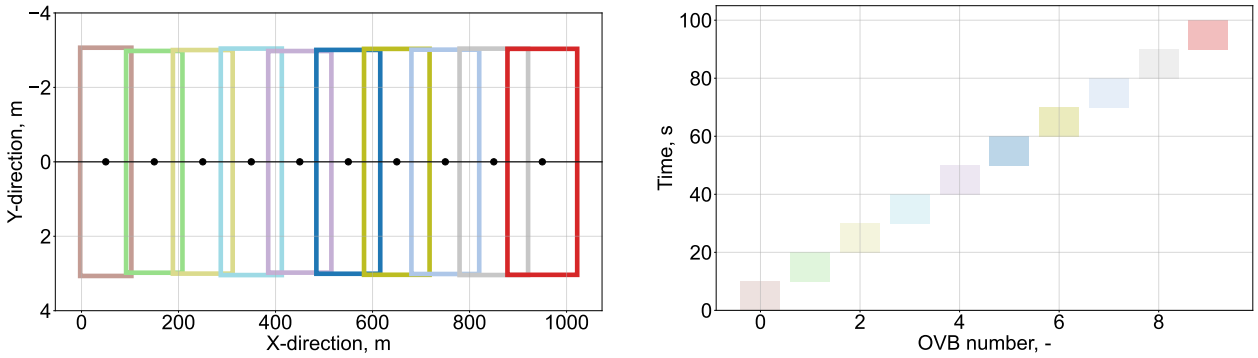
(a) Spatial dimension 2D top-down view *(b) Temporal dimension*

Figure 11. *Length-fixed OVB sizing with temporal overlap.*

Similarly, Figure 12 shows the same 10 OVBs when their duration is fixed at 10 seconds. The trajectories are omitted for clarity. In this case, the emerging overlap is spatial rather than temporal. OVB length and the spatial overlap increase linearly with time due to the longitudinal drift.

Regardless of the method used, it was observed that without PSO, the achieved conformance rate fluctuated between 94.8% and 95.1% closely aligning with the target of 95%. When PSO was applied, the target was consistently met; however, this came at the cost of a 4–12x increase in

[‖]github.com/astm-utm/Protocol/blob/F3548-21/utm.yaml



(a) Spatial dimension 2D top-down view *(b) Temporal dimension*

Figure 12. *Duration-fixed OVB sizing with spatial overlap.*

computation time. Therefore, in scenarios where the operator must periodically update the OVB dimensions for multiple aging vehicles based on their flight data, PSO fine-tuning may not be justified, given its marginal performance improvement relative to significant computational overhead.

Table 6 summarizes the average dimensions input to the flight path discretization module for both, length-fixed (LF) and duration-fixed (DF) OVBs. Given the cruising speed of 10 [m/s], the temporal overlap in the LF case corresponds to approximately 32 [m] of spatial overlap, closely matching the 31.5 [m] overlap in the DF case. This indicates that both approaches provide nearly equivalent safety margins. By extending the LF OVBs in both directions by half the temporal overlap (in distance), the DF OVBs naturally emerge. Nevertheless, this observation should be further studied through more extensive simulations and real-world data validation.

Table 6. *OVB dimensions for both sizing input types*

	Length [m]	**Width [m]**	**Height [m]**	**Duration [s]**	**Spatial overlap [m]**	**Temporal overlap [s]**
LF	100	6.1	1.2	13.3	0	3.2
DF	129.3	6.0	1.2	10.0	31.5	0

If the hypothesis is true, the choice between spatial and temporal overlap in OVBs ultimately depends on implementation details and operator preference. Spatial overlap makes the safety margin visually explicit, but it may clutter the display; in contrast, temporal overlap is less visually intrusive but renders the margin less apparent. Alternatively, both overlap types could be combined by partially converting one form into the other using the cruise speed. A well-informed recommendation on the appropriate overlap type could be developed through future human factors studies that account for all relevant user interface variables.

3.2 Scenario Generation

For the current example, 50 flights cruise at the same altitude in the area of Mountain View, California: 10 HNS flights split equally among two depots from Company H0, 10 HNS flights split equally among two depots from Company H1, 10 ONB flights from Company O0 and 10 P2P flights

from Company P0. The inputs for each of the operational profiles can be found in Table A-1 in the Appendix. Figure 13 and Figure 15 show all the planned flights for the scenario spatially and temporally, respectively.

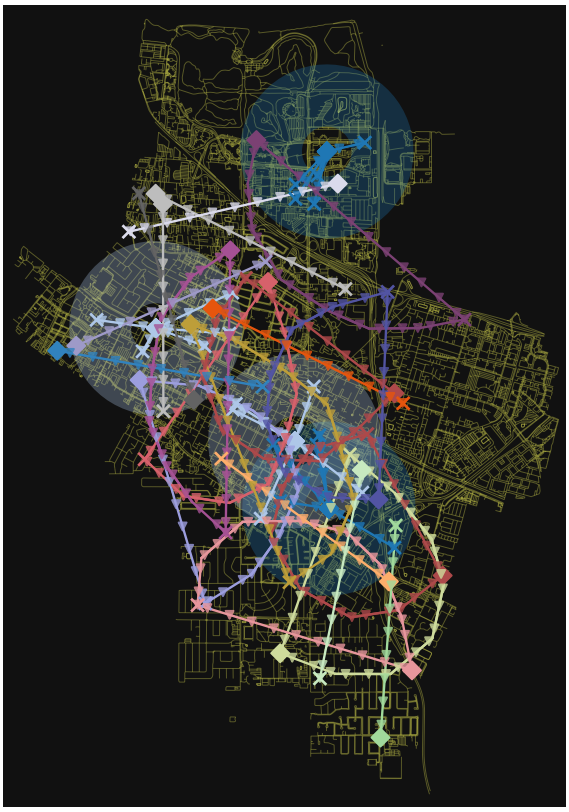


Figure 13. *Example scenario: Top-down view of airspace.*

Figure 14. *Example scenario: Flight path discretization with zoom-in regions.*

By leveraging real-world operational profiles tied to local infrastructure, a wide range of scenarios with realistic complexity can be automatically generated. This approach eliminates the need to manually design a limited set of airspace geometries, which may be either too simplistic or overly complex—and whose findings may be too narrowly tailored to specific cases. By periodically varying geographic locations and sampling operational parameters from realistic ranges or probability distributions, airspace diversity can be introduced into Monte Carlo studies. This enables more robust assessments of average safety and efficiency for UTM services, such as CMSA. As a result, conclusions can be drawn at scales ranging from individual neighborhoods to the entire National Airspace System (NAS).

3.3 Flight Path Discretization

For visualization purposes, the OVBs in the example scenario have been enlarged relative to their original dimensions from the sizing stage. To demonstrate the tool's versatility — allowing each company to use different OVB configurations — four variations are presented in Table A-2 in

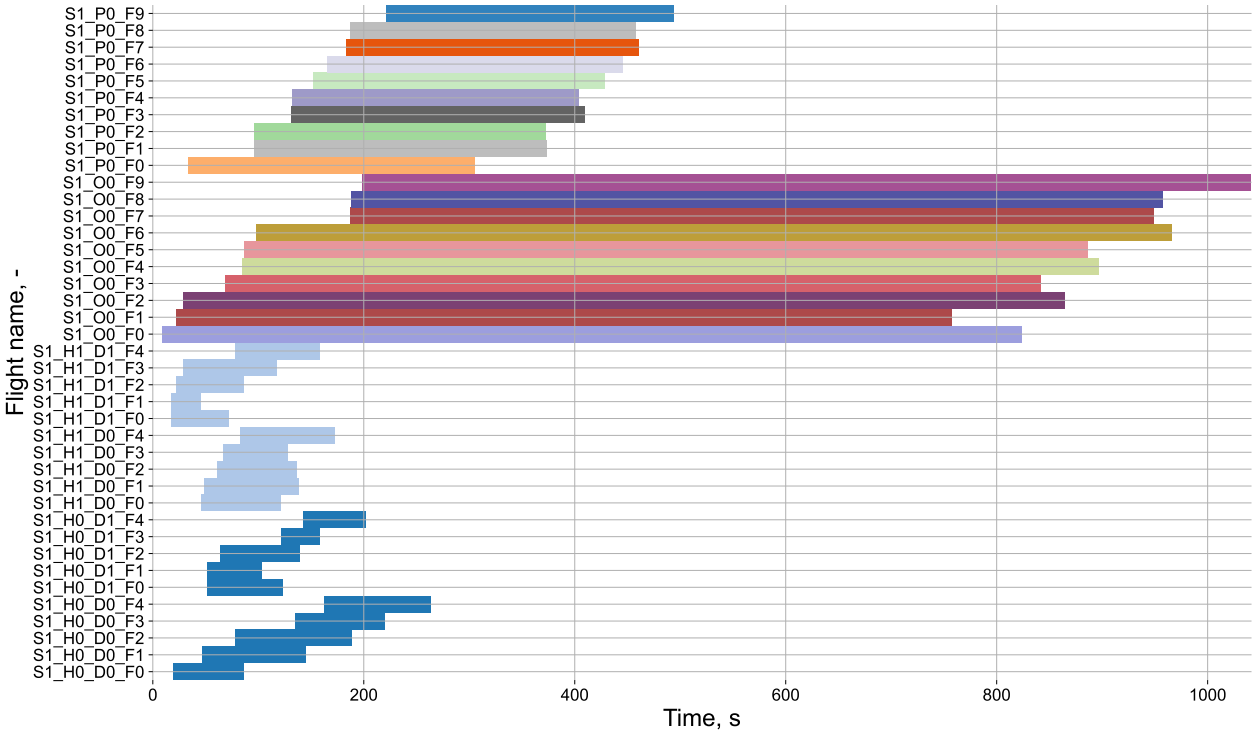


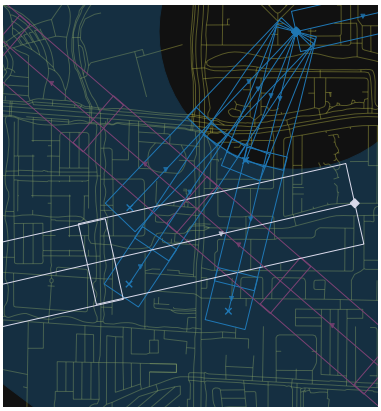
Figure 15. *Example scenario: Flight paths temporal plot. Flight names on the y-axis follow the format: S<scenario number>_<operational profile type><company number>_F<flight number>, where the service type is represented by a letter ('O' for ONB, 'H' for HNS, 'P' for P2P) and the flight number is assigned by each company. For HNS, an additional depot identifier is included in the format D<depot number>. The colors correspond to those shown in the airspace illustration.*

the Appendix: H0 serves as the baseline; H1 uses temporal overlap instead of spatial; P0 features OVBs twice the size of those in H0; and O0 maintains the same dimensions as H0 but includes additional width buffers. Figure 14 illustrates how the methodology in Section 2.3 applies those OVB dimensions to the example scenario.

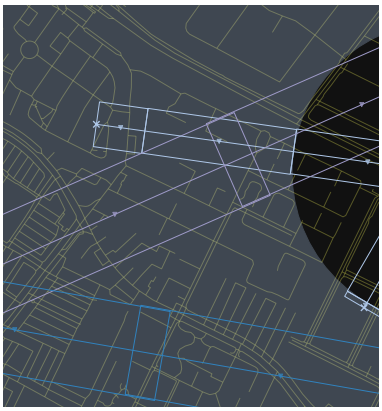
Figure 16 presents three zoomed-in views from Figure 14, highlighting specific spatial features. In Figure 16a, the spatial overlap among the three companies is shown, along with noticeable differences in length and width. P0 is twice the size of H0, while O0 is 1.5 times wider than H0 due to its buffer. Additionally, the H0 flights show that the base of the first OVBs aligns with the start of cruise above their depot. Figure 16b illustrates that H1's OVBs are tangential rather than overlapping, reflecting the absence of spatial overlap. These OVBs are also half the length and width of P0's. Finally, Figure 16c demonstrates how smaller OVBs are generated just before waypoints for O0 flights, while still preserving spatial overlap.

3.4 4D Volume-Based Conflict Detection

All R-tree 4D volume-based conflict detection implementations consistently identify the 93 conflicts shown in Figure 17. Since the primary goal is to use R-trees in fast-time simulation studies, and given their critical role in the conflict resolution strategy described in Section 2.5, the most suitable implementation is the one that yields the lowest computational time.



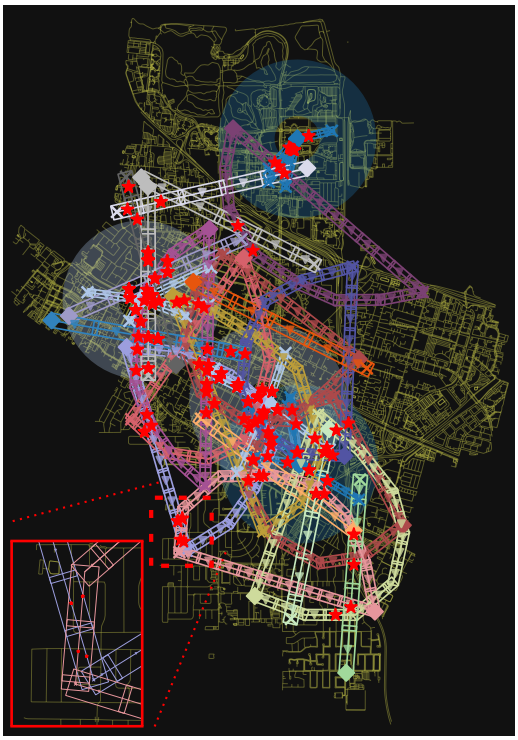
(a) *Top-right corner zoom-in with blue OVBs from H0 depot, red OVBs from O0 and white OVBs from P0*



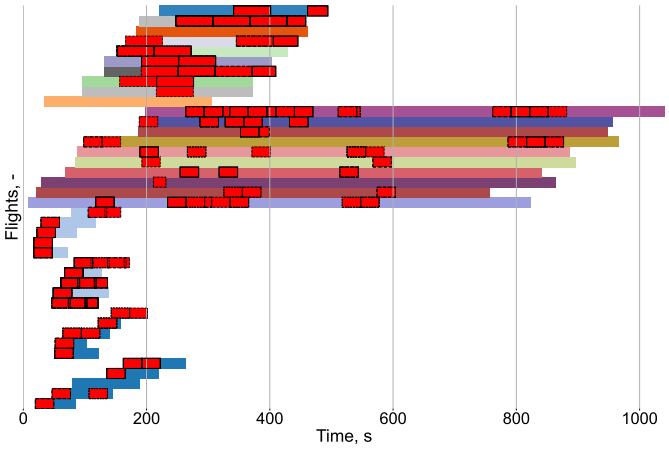
(b) *Center-left zoom-in with white OVBs from H1 depot, and blue and purple OVBs from P0*

(c) *Bottom-right zoom-in with red, yellow and pink OVBs from O0*

Figure 16. *Flight path discretization examples: Zoom-ins from Figure 14.*



(a) *Top-down view of airspace. Centers of conflicting OVBs are marked with red crosses. Zoom-in region is highlighted with a red dashed box.*

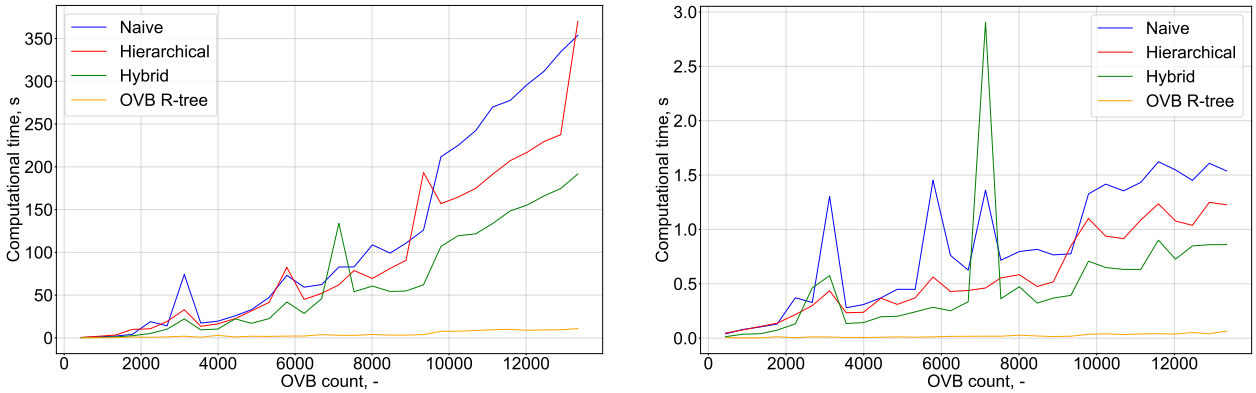


(b) *Flight paths temporal plot. The omitted vertical axis tick labels match those in Figure 15. Conflicting OVBs are in red.*

Figure 17. *Example scenario: 4D volume-based detected conflicts.*

To evaluate this, two key metrics were considered: (1) the time required to detect all conflicts among flights initially present in the airspace, and (2) the time needed to identify conflicts for a newly introduced flight, assuming all previous conflicts have already been computed and the R-tree is pre-built. The former is essential for computing the initial solution for the root node of the CT in CBS, while the latter is critical during execution of its low-level algorithm. To measure this, all R-tree implementations and the naive baseline performed each task with increasing numbers of flights initially present in the airspace. Each scenario is an equal mixture of each operational type.

As shown in Figure 18, incorporating any form of R-tree generally improves both metrics compared to the naive baseline. Even though the hierarchical and hybrid implementations might be more costly when initializing the scenario than the naive baseline, they offer performance gains every time a new flight is added. Among all variants, the OVB R-tree consistently outperforms the others, with its advantage growing steadily as the number of OVBs increases. This shows that the R-tree structure is the most effective when applied directly to geometric primitives (OVBs); abstractions (OIs) that separate them add overhead without benefit. Consequently, the OVB R-tree approach is selected for the low-level algorithm in the CBS 4D volume PDSTD framework.



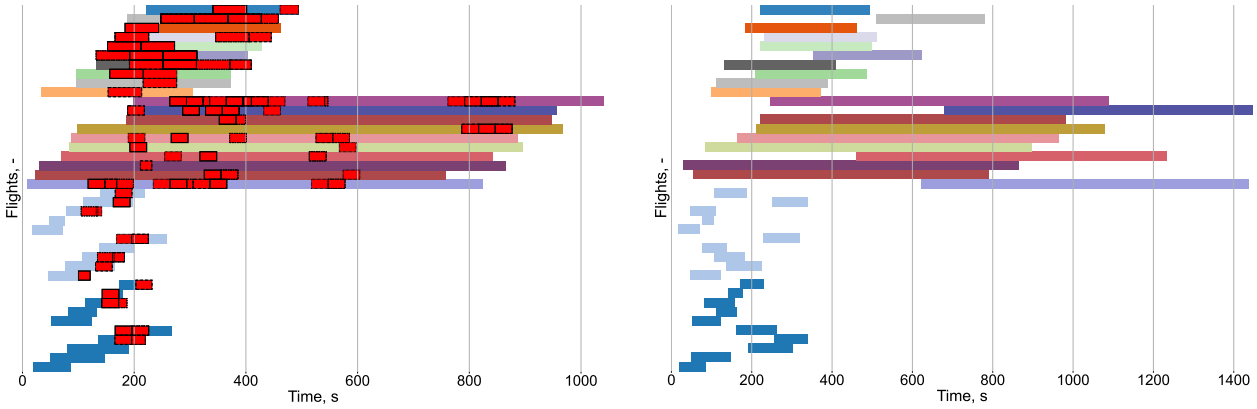
(a) *Initial scenario airspace* ***(b)*** *New flight added to deconflicted airspace*

Figure 18. *4D volume-based conflict detection computational time for R-tree implementations.*

3.5 4D Volume-Based Pre-Departure Strategic Temporal Deconfliction

As shown in Figure 17, many conflicts stem from overlapping HNS flight paths within depots, particularly when multiple vehicles take-off in rapid succession. This worsens with more depots and vehicles per depot. Commercial package delivery services, which often follow the HNS operational model, are frequent contributors due to their high flight volumes. Since CBS computational time scales with conflict count, reducing them leads to faster solutions. This can be done by temporally staggering the activation of OVBs from the same depot. In the example scenario (see Table A-2 in the Appendix), neither HNS company uses buffers, so increasing the minimum departure interval to 30 seconds, the duration of an OVB, reduces conflicts from 93 to 76, as shown in Figure 19a.

Regardless of whether HNS flights are staggered at the depot, the 4D volume-based PDSTD approach successfully resolves all conflicts by introducing delays to STDs of selected flights. Figure 19b presents the temporal plot of the resolved flight paths for the example scenario, which required a total induced delay of 3661.42 [s].



(a) *Flight paths temporal plot with temporally staggered HNS flights in each depot*

(b) *Deconflicted flight paths temporal plot*

Figure 19. *Example scenario: CBS 4D volume-based PDSTD. The omitted vertical axis tick labels match those in Figure 15. Conflicting OVBs are in red.*

Figure 20. *Example scenario: Top-down view of original airspace with added white flight on top-left.*

The proposed PDSTD algorithm resolved the 93 conflicts in 122.42 [s]. To evaluate the incremental runtime for deconflicting a single flight, an ONB flight with 30 waypoints was introduced into the already deconflicted airspace. Represented by the white OI in Figure 20, this flight generated 11 new conflicts, which were resolved by the algorithm in 3.5 [s].

Finally, due to the inherently disjoint nature of the CBS node solution computation, an attempt was made to parallelize the PDSTD algorithm using multiprocessing. Two approaches were tested: (1) assigning a single node to each worker and (2) assigning batches of nodes per worker. Using 8 workers and batches of 5 nodes, the results (see Table 7) show that the parallel implementation results in higher computation time. This indicates that the implicitly introduced beam search strategy, where the top k candidate nodes are expanded, can outperform the greedy strategy, which selects only the best node at each iteration. However, since the module is intended for fast-time simulations, the sequential approach remains the default.

Table 7. *CBS parallelization performance. PX denotes the number of X parallel processes, while BZ represents the batch size Z.*

	Final delay [s]	**Computational time [s]**	**# explored nodes**
P1	3661.42	122.42	525
P8	3307.33	218.43	771
P8B5	2594.71	948.40	2427

4 Conclusions

This paper proposes a set of simulation modules to enable future simulation studies that evaluate UAS Traffic Management (UTM) services (e.g., Conformance Monitoring for Situational Awareness or CMSA) on metrics that, among other aspects, seek a safe and efficient airspace. The modules were Operational Volume Block (OVB) sizing, flight path generation, flight path discretization, 4D volume-based conflict detection, 4D volume-based pre-departure strategic temporal deconfliction and a scenario exporter.

In OVB sizing, fixing the OVB blocks in length or durations has led to the natural emergence of temporal and spatial overlap which are associated with consecutive and overlapping OVBs, respectively. It was shown that there are signs that the emerging spatial and temporal overlaps are equivalent. Additionally, the added value of fine-tuning the OVB dimensions with Particle Swarm Optimization was observed to be very small.

Instead of tweaking the airspace to reach certain complexity levels, realistic scenarios are automatically created. This is achieved by randomly and periodically selecting geographic locations and spawning services that follow operational profiles that have been linked to actual geographic features. Consequently, virtually infinite scenarios can be simulated and conclusions can be drawn at scales ranging from individual neighborhoods to the entire National Airspace System (NAS).

R-trees constructed using all individual airspace OVBs consistently outperform both the baseline and other R-tree implementations considered for 4D volume-based conflict detection, with its advantage growing steadily as the number of OVBs increases. This demonstrates that the R-tree structure is most effective when applied directly to geometric primitives, whereas higher-level abstractions, such as Operational Intents (OIs), introduce unnecessary overhead without providing performance benefits.

The proposed Conflict-Based Search for 4D volume-based pre-departure strategic temporal deconfliction successfully resolves all conflicts in the scenario, regardless of intersection geometry (merging, crossing, parallel). Whereas the beam search strategy, implicitly introduced through algorithm parallelization, yields lower total airspace delays, the default greedy strategy outperforms it in terms of computational time, which is the primary consideration in fast-time simulations.

Further work starts with the development of a factory of vehicle failures, such as propeller damage (Ref. 41, 42), that can be injected in the simulation to create off-nominal flights, as well as the CMSA modules illustrated in Figure 1. Once the failure has been injected, the CMSA module should be able to detect the off-nominality, transition the states of the vehicle (e.g., per the ASTM standards (Ref. 21)) and create the predicted off-nominal OVBs used for the deconfliction.

Moreover, it is necessary to validate the OVB sizing methodology with real flight data and verify the hypothesis that the spatial and temporal overlaps are equivalent in terms of safety. In terms of flight path generation, the out-and-back and point-to-point operational profiles could be made more realistic by connecting them to geographical features, and other existing and emerging operational profiles should be implemented. Furthermore, it should be studied whether the 4D volume-based pre-departure strategic temporal deconfliction of the initial scenario airspace could be accelerated by (1) first deconflicting each depots in isolation and (2) introducing a heuristic, like the rolling horizon, to divide the problem space further. The implementation and safety benefit assessment of in-flight strategic deconfliction and tactical deconfliction (Ref. 43) should be considered.

Even though the fast-time simulation blocks presented here do not address the implementation of UTM services, such as CMSA, they provide a critical foundation for translating the information generated by these services into safety benefits applicable across the future NAS. The authors aim for this work to serve as a first step towards relieving researchers of the repetitive but essential tasks of scenario generation and initial deconfliction. This would enable greater focus on the development and analysis of current and more advanced UTM functionalities that will shape future technical requirements and regulations.

Appendix

Table A-1 contains the input parameters used to define the operational profiles used in the example scenario and Table A-2 shows the OVB dimensions of each company.

Table A-1. *Operational profile parameters of the example scenario (see Section 3)*

	HNS	**ONB**	**P2P**
Number of companies	2	1	1
Number of depots per company	2	-	-
Number of flights per depot (HNS) or per company (ONB/P2P)	5	10	10
Depot tags	restaurant, bar, cafe, pub, supermarket, shop	-	-
Client tags	residential, hotel, office, college, university, stadium, dormitory, static caravan, stilt house, house, bungalow	-	-
Intra-company depot distance	2000	-	-
Min. clients per depot	30	-	-
Max./Min. radius of operations	1000/300	-	-
Depot min. temporal flight separation	0	-	-
Max./Min. number of waypoints	-	7/6	-
Max./Min. inter-waypoint distance	-	700/500	-
Waypoint max. heading change	-	45	-
Waypoint (ONB) or take-off/landing (P2P) search limit	-	50	50
Turn single direction flag	-	1	-
Max./Min. flight distance	-	-	2500/2400
Average take-off rate	20	20	20

Table A-2. *OVB dimensions for each company in example scenario (see Section 3)*

	l [m]	w [m]	h [m]	t [s]	O_s [m]	O_t [s]	**Buffers** ***l-w-h-t***
H0	300	90	60	30	30	0	0-0-0-0
H1	300	90	60	30	0	3	0-0-0-0
P0	600	180	120	60	60	0	0-0-0-0
O0	300	90	60	30	30	0	0-45-0-0

References

1. Gaede, V.; and Günther, O.: Multidimensional Access Methods. *ACM Comput. Surv.*, vol. 30, no. 2, June 1998, p. 170–231.

2. Federal Aviation Administration: The Economic Impact of U.S. Civil Aviation in 2020. 2022-APL-038, Federal Aviation Administration, August 2022.

3. Federal Aviation Administration: FAA Aerospace Forecast Fiscal Years 2025–2045. U.S. Department of Transportation, Federal Aviation Administration, June 2025.

4. Mohd Daud, S. M. S.; Mohd Yusof, M. Y. P.; Heo, C. C.; Khoo, L. S.; Chainchel Singh, M. K.; Mahmood, M. S.; and Nawawi, H.: Applications of drone in disaster management: A scoping review. *Science & Justice*, vol. 62, no. 1, 2022, pp. 30–42.

5. Akhloufi, M. A.; Couturier, A.; and Castro, N. A.: Unmanned Aerial Vehicles for Wildland Fires: Sensing, Perception, Cooperation and Assistance. *Drones*, vol. 5, no. 1, 2021.

6. Chakrabarty, A.; and Ippolito, C. A.: Wildfire monitoring using Unmanned Aerial Vehicles operating under UTM (STEReO). *AIAA Scitech 2021 Forum*, 2021.

7. Ellis, K.; Johnson, M.; Neogi, N.; and Homola, J.: NASA Research to Expand UAS Operations for Disaster Response. *34th Congress of the International Council of the Aeronautical Sciences (ICAS)*, Florence, Italy, September 2024. NASA Technical Reports Server (NTRS) Accession Number: NASA/TM-20240007804.

8. U.S. Department of Transportation and National 911 Program: The Uses Continue to Emerge: Public Safety Drones and Considerations. 911.gov, 2023. URL `https://www.911.gov/assets/Drones_and_Public_Safety.pdf`, accessed: 2025-07-23.

9. Harriss, D.; Sheh, R.; and Geappen, K.: Autonomous Aerial Drones Connecting Public Safety: Opportunities and Challenges for the Future. *Exponential 2024*, September 2024.

10. Ramon-Soria, P.; Perez-Jimenez, M.; Arrue, B. C.; and Ollero, A.: Planning System for Integrated Autonomous Infrastructure Inspection using UAVs. *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2019, pp. 313–320.

11. Yu, L.; Yang, E.; Ren, P.; Luo, C.; Dobie, G.; Gu, D.; and Yan, X.: Inspection Robots in Oil and Gas Industry: a Review of Current Solutions and Future Trends. *2019 25th International Conference on Automation and Computing (ICAC)*, 2019, pp. 1–6.

12. Rejeb, A.; Abdollahi, A.; Rejeb, K.; and Treiblmaier, H.: Drones in agriculture: A review and bibliometric analysis. *Computers and Electronics in Agriculture*, vol. 198, 2022, p. 107017.

13. Choudhury, S.; Solovey, K.; Kochenderfer, M. J.; and Pavone, M.: Efficient Large-Scale Multi-Drone Delivery Using Transit Networks. *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 4543–4550.

14. Johnson, A.; Cunningham, C.; Arnold, E.; Rosamond, W.; and Zègre-Hemsey, J.: Impact of Using Drones in Emergency Medicine: What Does the Future Hold? *Open Access Emerg Med*, vol. 13, November 2021, pp. 487–498.

15. National Aeronautics and Space Administration and Federal Aviation Administration: UAS Traffic Management (UTM) Concept of Operations, Version 2.0. Federal Aviation Administration, Washington, DC, March 2020.

16. Homola, J.; Dao, Q.; Martin, L.; Mercer, J.; Mohlenbrink, C.; and Claudatos, L.: Technical capability level 2 unmanned aircraft system traffic management (UTM) flight demonstration: Description and analysis. *2017 IEEE/AIAA 36th Digital Avionics Systems Conference (DASC)*, 2017, pp. 1–10.

17. Homola, J.; Martin, L.; Cencetti, M.; and Aweiss, A.: UAS Traffic Management (UTM) Technical Capability Level 3 (TCL3) Flight Demonstration: Concept Tests and Results. *2019 IEEE/AIAA 38th Digital Avionics Systems Conference (DASC)*, 2019, pp. 1–8.

18. Martin, L.; Wolter, C.; Jobe, K.; Manzano, M.; Blandin, S.; Cencetti, M.; Claudatos, L.; Mercer, J.; and Homola, J.: TCL4 UTM (UAS Traffic Management) Nevada 2019 Flight Tests, Airspace Operations Laboratory (AOL) Report. Technical Memorandum NASA/TM-2020-5003361, NASA Ames Resscearch Center, Moffett Field, CA, March 2020.

19. Smith, H.: Learn More About NASA's UTM BVLOS Subproject. `https://www.nasa.gov/directorates/armd/aosp/atm-x/utm-bvlos/about-utm-bvlos/`, December 2024. Accessed: 2025-07-23.

20. Goldstein, B.: FAA Approves Wing, Zipline BVLOS Drone Ops In Dallas. `https://aviationweek.com/advanced-air-mobility-departments/faa-approves-wing-zipline-bvlos-drone-opsdallas`, July 2024. Accessed: 2025-07-23.

21. ASTM Committee F38 on Unmanned Aircraft Systems: Standard Specification for UAS Traffic Management (UTM) UAS Service Supplier (USS) Interoperability. ASTM International, 2021.

22. Evans, A. D.; Egorov, M.; Anand, A.; Campbell, S. E.; Zanlongo, S.; Young, T.; and Sarfaraz, N.: Safety Assessment of UTM Strategic Deconfliction. *AIAA SciTech 2023 Forum*, 2023.

23. Xue, M.; Kuo, V. H.; d. de Alvear Cárdenas, J. I.; and Pradeep, P.: Safety Benefit Analysis of Conformance Monitoring for Situation Awareness in UTM. *AIAA SciTech 2025 Forum*, 2025.

24. Kuo, V. H.; Xue, M.; Pradeep, P.; de Alvear Cárdenas, J. I.; and Lee, S.: Safety Assessment of Conformance Monitoring for Situational Awareness in UTM Operations. Technical Memorandum NASA/TM-20240015613, NASA Ames Research Center, December 2024.

25. de Alvear Cárdenas, J. I.; Pradeep, P.; Xue, M.; Lee, S.; and Kuo, V. H.: Confidence-Based Buffer for Strategic Deconfliction With Probabilistic Operational Intent. *AIAA SciTech 2025 Forum*, 2025.

26. Pradeep, P.; Lee, S.; Silva, J. P.; de Alvear Cárdenas, J. I.; Kuo, V. H.; Xue, M.; and Yarramreddy, G. S.: Simulation Studies of 4D Volume-based Pre-departure Strategic Deconfliction under Wind Uncertainties for Package Delivery sUAS. Technical Memorandum NASA/TM-20250005293, NASA Ames Research Center, May 2025.

27. Guttman, A.: R-trees: a dynamic index structure for spatial searching. *Proceedings of the 1984 ACM SIGMOD International Conference on Management of Data*, SIGMOD '84, Association for Computing Machinery, New York, NY, USA, 1984, p. 47–57.

28. Sharon, G.; Stern, R.; Felner, A.; and Sturtevant, N. R.: Conflict-based search for optimal multi-agent pathfinding. *Artificial Intelligence*, vol. 219, 2015, pp. 40–66.

29. Xue, M.; Rios, J.; Silva, J.; Zhu, Z.; and Ishihara, A. K.: Fe^3: An Evaluation Tool for Low-Altitude Air Traffic Operations. *2018 Aviation Technology, Integration, and Operations Conference*, 2018.

30. Verma, S. A.; Monheim, S. C.; Moolchandani, K. A.; Pradeep, P.; Cheng, A. W.; Thipphavong, D. P.; Dulchinos, V. L.; Arneson, H.; Lauderdale, T. A.; Bosson, C. S.; Mueller, E. R.; and Wei, B.: Lessons Learned: Using UTM Paradigm for Urban Air Mobility Operations. *2020 AIAA/IEEE 39th Digital Avionics Systems Conference (DASC)*, 2020, pp. 1–10.

31. Hsieh, C.; Sibai, H.; Taylor, H.; Ni, Y.; and Mitra, S.: SkyTrakx: A Toolkit for Simulation and Verification of Unmanned Air-Traffic Management Systems. *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, 2021, pp. 372–379.

32. Egorov, M.; Evans, A.; Campbell, S.; Zanlongo, S.; and Young, T.: Evaluation of UTM strategic deconfliction through end-to-end simulation. *14th USA/Europe Air Traffic Management Research and Development Seminar (ATM2021)*, 2021, pp. 20–23.

33. Xue, M.; Kuo, V. H.; de Alvear Cárdenas, J. I.; and Pradeep, P.: A Method of Compliance for Achieving Target Collision Risk in UTM Operations. Technical Memorandum NASA/TM-20240003151, NASA Ames Research Center, February 2024.

34. Pradeep, P.; Munishkin, A. A.; Kalyanam, K. M.; and Erzberger, H.: Strategic Deconfliction of Small Unmanned Aircraft Using Operational Volume Blocks at Crossing Waypoints. *AIAA SciTech 2023 Forum*, 2023.

35. Shirazi, M. J.; Vatankhah, R.; Boroushaki, M.; Salarieh, H.; and Alasty, A.: Application of Particle Swarm Optimization in Chaos Synchronization in Noisy Environment in Presence of Unknown Parameter Uncertainty. *Communications in Nonlinear Science and Numerical Simulation*, vol. 17, no. 2, 2012, pp. 742–753.

36. de Alvear Cárdenas, J. I.; Sun, B.; and Kampen, E.-J. V.: Intelligent Adaptive Control Using LADP and IADP Applied to F-16 Aircraft with Imperfect Measurements. *AIAA Scitech 2021 Forum*, 2021.

37. Boeing, G.: Modeling and Analyzing Urban Networks and Amenities With OSMnx. *Geographical Analysis*, 2025.

38. Pradeep, P.; Yarramreddy, G. S.; Amirsoleimani, N.; Munishkin, A.; Morris, R. A.; Xue, M.; Chour, K.; and Kalyanam, K.: Rolling Horizon With K-Position Search Method for Strategic Deconfliction of Package Delivery UAS. *AIAA Aviation Forum and Ascend 2024*, 2023.

39. Huang, J.; Bao, G.; and Li, Q.: Realization of R-tree for GIS on Hybrid Clustering Algorithm. *Journal of Central South University of Technology*, vol. 12, 2005, pp. 601–605.

40. Zhu, Q.; Gong, J.; and Zhang, Y.: An Efficient 3D R-tree Spatial Index Method for Virtual Geographic Environments. *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 62, no. 3, 2007, pp. 217–224.

41. de Alvear Cárdenas, J. I.; and de Visser, C. C.: Blade Element Theory Model for UAV Blade Damage Simulation. *AIAA SciTech 2024 Forum*, 2024.

42. de Alvear Cárdenas, J. I.; and de Visser, C. C.: Unreal Success: Vision-Based UAV Fault Detection and Diagnosis Framework. *AIAA SciTech 2024 Forum*, 2024.

43. Yarramreddy, G. S.; de Alvear Cárdenas, J. I.; Pradeep, P.; Xue, M.; Lee, S.; and Kuo, V. H.: Simulation Framework for Tactical Separation Assurance Service using Deep Reinforcement Learning. Technical Memorandum NASA/TM-2025-0002761, NASA Ames Research Center, Moffett Field, CA, March 2025.