Generic Urban Air Mobility Simulation

Michael J. Acheson*, Andrew Patterson†, and Irene M. Gregory‡
NASA Langley Research Center, Hampton, VA, 23681

This research presents a simulation framework for autonomous research for a UAM vehicle using the NASA Revolutionary Vertical Lift Technology Lift+Cruise concept vehicle. Our research results were produced using the open-source, six degree of freedom, rigid-body, non-linear generic urban air mobility (GUAM) simulation. The intent of this paper is to demonstrate the GUAM simulation and a series of Challenge Problems that our researchers have posed to the broader autonomous vehicle research community. Our team has developed the GUAM simulation for the express purpose of providing a high-fidelity transition vehicle dynamics model to foster collaboration and algorithm performance comparison across research teams. In this paper, we demonstrate some of the autonomous flight research challenges and some of our current approaches to tackling basic autonomous flight tasks (e.g., trajectory following, stationary and moving obstacle avoidance). Additionally, we propose some flight metrics to assess autonomous algorithm performance while accomplishing these basic autonomous tasks.

I. Introduction

The aerospace transportation ecosystem is currently undergoing a revolution with the incorporation of a wide array of small, electrified transition vehicles. The business case for the advanced air mobility (AAM) and urban air mobility (UAM) paradigms heavily favor fully autonomous vehicle operations. However, the gaps between the current state of autonomous vehicle operations and full autonomy are substantial. One needs to look no further than the state of automobile autonomy (which is a simplified 2-D problem with defined paths, control signals/signs, and rulesoftheroad) to recognize that full aerospace autonomy requires many future advances. Furthermore, autonomous automobiles can simply stop to try to prevent accidents, which is a luxury that aerospace vehicles often do not have. The gaps in autonomy can be broadly categorized as: obstacle/vehicle detection and perception, real-time trajectory replanning, real-time vehicle performance modeling and flight envelope prediction, fault identification, strategic planning and decision making, and contingency management. The unique designs of UAM vehicles present issues of aero-propulsive modeling, and flight control of transition vehicles which further complicate autonomy. As such, the design, certification, and operation of autonomous transition vehicles touches a wide array of research specialties. These include the classical aerospace fields of structures, propulsion, aerodynamics, aeroelasticity, aeroacoustics, and controls. However, the integrated aero-propulsive vehicle configurations (e.g., multiple rotors/propellers, tilt-rotors, and tilt-wings) are not well understood due to the substantial propulsor-to-propulsor and propulsor-to-airframe interactions. The novel electrified propulsor hardware systems are still being designed and developed with all the growing pains of new systems, which implies increased system failure rates compared to legacy aircraft systems. Moreover, the limitations of battery/electric power generation systems dictate very limited vehicle performance margins when compared with legacy (fuel-powered) vehicles. Beyond the typical aerospace research areas, autonomous vehicles require autonomous perception and situational awareness. These skills have historically been the domain of robotic systems in the computer science and machine learning fields. Additionally, cognitive processes and decision making have similarly been in the computer science, machine learning, and artificial intelligence fields, which are difficult to develop and deploy with aerospace systems. Real-time trajectory replanning has been widely researched. However, the replanning process must be fused with both tactical and strategic vehicle objectives, and incorporate contingencies (e.g., weather, obstacles, system failures), all while dealing with substantial vehicle performance margin limitations. Furthermore, the field of human-machine interface is required both at the vehicle and the aerospace systems level. These vehicles are not being developed in a vacuum, as there is an existing aerospace ecosystem based on human pilots. Therefore, autonomous vehicles will be required to operate within completely human-centric, hybrid, and fully autonomous vehicle operational environments.

*Senior Research Engineer, Dynamic Systems and Control Branch, Michael.J.Acheson@nasa.gov, Member AIAA.
†Research Engineer, Dynamic Systems and Control Branch, Andrew.Patterson@nasa.gov.
‡NASA Senior Technologist for Advanced Control Theory and Application, Irene.M.Gregory@nasa.gov, Fellow AIAA.

It was recognized by our research team that the diverse array of research fields, coupled with the limited budgets, staffs, and expertise, necessitate collaboration and cooperation between research groups if the goal of full autonomous transition vehicle operations is to be achieved. Many barriers exist that hinder collaboration between industry, academia, and government. Industry possesses the most access to real-world transition flight vehicle systems, but there are very substantial issues with intellectual property rights that must be overcome. Moreover, industry inherently possesses limited staffing and a profit focus that limits basic research in an effort to field a product as soon as possible. Conversely, academia possesses a substantial pool of staff (i.e., graduate students and post-docs), but this is a rotating staff which is required, by nature, to produce novel research and is generally not focused on technology readiness level (TRL) development. Moreover, while academia generally has the least restrictive access to small flight vehicles, such as small quadrotor, fixed wing, and helicopter platforms, it is difficult for them to access larger platforms to scale their results. Finally, the government's research focus is generally both basic research and longer-term development, but it is not generally focused on TRL and product development. Government faces substantial barriers to flight operations but does have access to a wide array of relevant vehicles. Additionally, government research staffs in each particular area are small, but budgets are periodically available to fund outside research (e.g., NASA grants and cooperative agreements).

To address the autonomous aerospace vehicle challenges and barriers to collaboration noted previously, our research team has developed a high-fidelity, open-source, six degree of freedom, rigid-body, non-linear transition vehicle framework known as the generic urban air mobility (GUAM) simulation `https://github.com/nasa/Generic-Urban-Air-Mobility-GUAM`. This framework was designed from its conception to provide a complete aerospace-focused architecture (e.g., existing bus structures contain most frames of reference and aerospace quantities of interest) that would obviate the need for users to spend valuable resources on the creation and development of a simulation. It is intended that researchers can instead focus on their particular algorithm development/testing and simply drop in their research as systems into the existing simulation framework. Moreover, while this simulation is currently limited to two transition vehicles, the framework is readily adaptable for incorporation of other vehicles. The simulation and existing vehicles are open, and as such, any research is publishable without undue restrictions. The simulation was developed in Matlab® and Simulink®, but it is fully C/C++ auto-code capable. Our cognitive architecture incorporates Python®, and the simulation interfaces with external graphics capabilities (e.g., Unreal Engine and AirSim). Finally, the simulation includes multiple data sets which form the basis of a series of challenge problems we propose to the research community. In this way, researchers can use a common simulation framework and data sets to collaborate, baseline results, and compare algorithm performance across the many research areas necessary to advance autonomous flight. More information on the GUAM autonomy challenge problems can be found on the NASA TTT Revolutionary Aviation Mobility (RAM) site `https://nari.arc.nasa.gov/ttt-ram/community`. In this paper, Sec. II discusses autonomy in aviation, and Sec. III details the challenge problems and associated data sets provided with GUAM. Section IV demonstrates some basic autonomy tasks and discusses metrics using the challenge problem datasets. Finally, Sec. V provides some closing remarks and outlines future research.

II. Autonomy in Aerospace

Currently in aviation, aerospace vehicles can readily demonstrate autonomy for nominal operations such as takeoff, cruise, and landing (MQ-25, X-45A, ILS) [1–3]. However, aerospace autonomy begins to show major gaps when encountering off-nominal or contingency events. Some basic contingencies in aviation occur on a routine basis such as: requirement for flight route (plan) changes, weather impacts on operations, traffic, non-catastrophic system failures, etc. However, more serious contingencies also occur on a fairly routine basis, such as, fuel emergencies, cascading and insidious systems failures, bird strikes, flight control problems, tire and/or landing gear issues. For human aviators, response actions to contingencies are prioritized using the mantra: aviate, navigate, and communicate. Aviate also typically implies that the first priority in a contingency is to maintain control of the vehicle (i.e., keep it upright). Aviate typically involves diagnosing the issue, and then adjusting systems and controls to keep the vehicle under control. The second priority for human response to contingency management is to navigate. Here, the human aviator, determine the best navigation plan in response to the emergency or at a minimum get the aircraft "nose pointed in the right direction," while continuing to address the contingency. The final priority for humans is to communicate, which typically entails communicating the nature of the issue and aviator intentions to the National Airspace System (NAS). Failures of humans to respond correctly to contingencies are very well documented [4, 5], but literature also shows examples of cases where human actions were highly successful in handling contingency situations (e.g., Captain Sullenberger on the Hudson river, United Airlines Flight 232 Sioux City) [6, 7]. The overall impression when looking at aviation incidents is to prescribe "pilot error" to most if not all serious contingencies. However, the truth is that human aviators are outstanding

at handling contingencies and in preventing them from ever occurring [8, 9]

So with a historical understanding of both the current state of aerospace autonomy and the human approach to contingencies, we make the following two assumptions. First, aerospace autonomy needs better tools and techniques to autonomously handle contingencies. Second the goal of autonomy in aviation is to replicate, and where possible, improve on human intelligent contingency management. It is therefore informative to ask what the aviate, navigate, and communication mantra translates to for autonomous systems. Figure 1 shows some of the wide array of autonomous capabilities needed to emulate human contingency priorities.

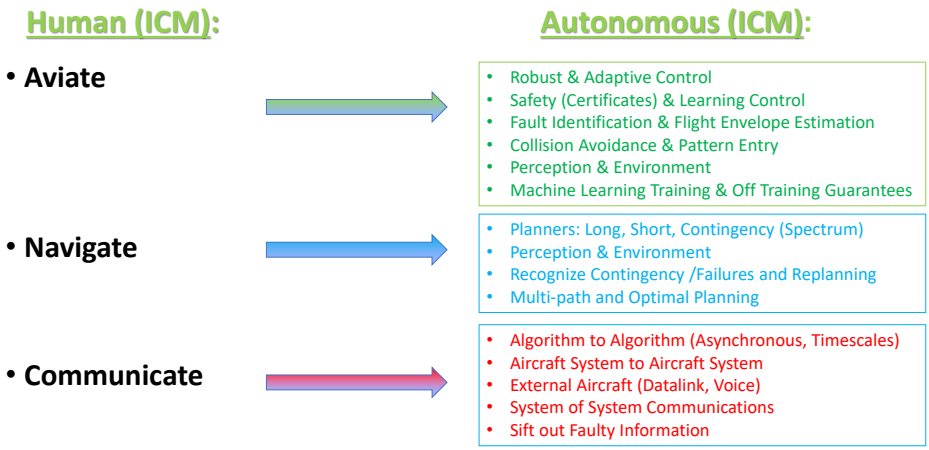


Fig. 1 Autonomous Intelligent Contingency Management

Most of the autonomous capabilities listed in Fig. 1 are entire research disciplines unto themselves. With this wide diversity of autonomy research disciplines, and the barriers to research groups (government, industry, and academia) outlined in section I, it was decided to formulate and release sets of autonomy challenge problems for use with the GUAM v1.1 open-source simulation.

III. Autonomy Challenge Problems

In this paper, autonomy challenge problems (also referred to as challenge problems) are intended to provide the diverse research teams working across different aerospace disciplines, the ability to utilize a common, open-source simulation framework for testing their autonomy algorithms at the level appropriate for their team's skill and expertise. The development and open-source release of GUAM v1.1 focused on providing a common framework with the tools necessary to challenge autonomy researchers. For instance, GUAM provides researchers access to a relevant Lift+Cruise transition vehicle with a unified (across transition phases) Robust Servomechanism Linear Quadratic Regulator (RSLQR) controller [10], atmospheric model (1976 atmosphere model with winds/turbulence), two aero-propulsive vehicle models, and the capability to emulate a host of relevant control effector failures. Additionally, GUAM v1.1 provides four sets of challenge problem data consisting of 3000 runs containing own-ship flight plans/dynamically feasible trajectories, randomized stationary obstacles, randomized moving obstacles, and randomized control effector (both aerodynamic and propulsive) failures. By providing the common simulation framework, and a rich set of simulation features along with scenario data sets, research teams can craft autonomy challenges appropriate to their respective research. Moreover, research teams can increase the level of autonomy difficulty by utilizing the simulation features and data sets in various combinatorial ways. For example, a research robust controller could be used to simply quantify the controller's ability to track the 3000 dynamically feasible trajectories. However, an entire array of research challenges can be examined (for increasing levels of challenge problem difficulty) by incrementally adding: winds/turbulence, aero-propulsive uncertainty (i.e., using one model to design the controller, but use the other model for the as-flown truth model), effector model uncertainty/time lags, implementing inflight stationary obstacles, combining stationary and moving obstacles in a scenario, and introducing arbitrary effector failures during flight. In this manner, the simulation and data sets

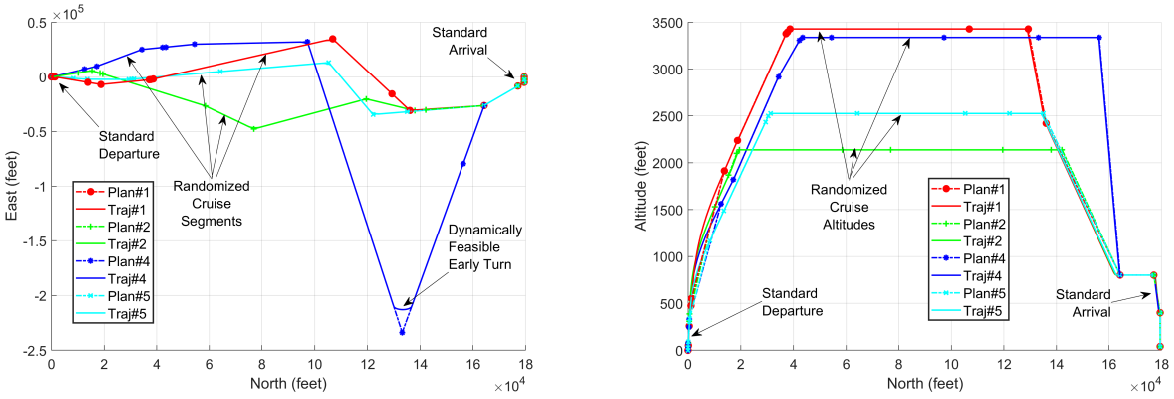
provide researchers complete freedom to focus on their particular research discipline while being able to add additional autonomy contingencies at will. In the computer science/machine learning disciplines, challenge problems typically refer to a defined problem (with the necessary code/data) along with metrics of performance such that teams compete (are ranked). In our challenge problem environment, we intend to present future specific autonomy challenges, such as detect and avoid, using photo realistic simulation (e.g., Unreal Engine), non-towered traffic pattern entry, etc., but our team will not be judging researcher's performance. Instead, the judging of autonomy challenge performance will be crowd-sourced using conference/journal publications, citations, and follow-on research funding. Moreover, it is our expectation that generating a body of autonomy research literature and code base (e.g., github) will allow researchers to readily baseline their work against their peers, foster easy collaboration, and facilitate straightforward performance comparison using the same vehicle model, simulation and data sets. Our team expects (and have already seen) a host of forks off the GUAM v1.1 github repo (e.g., Python® GUAM). The hope is that having a common framework to create and share research will advance the state-of-the-art in autonomy research.

A. Challenge Problem Datasets

This section describes in greater detail the creation and content of the data sets provided with the challenge problem folder within GUAM (`Challenge_Problems` folder). In general, these data sets were created with a dedicated Matlab® m-script for each of the four data sets. In this manner, researchers can recreate the data corresponding to the supplied .mat file, modify the scripts to generate additional data, and/or modify scenarios as they see fit.

Data set one contains two sets of 3000 own-ship trajectories. This dataset set was generated by the script `Generate_Own_Traj.m`. The first of the two types of own-ship trajectories (named `own_traj_orig`) is a basic flight plan consisting of a series of waypoints (i.e., time, positions). The second set of own-ship trajectories (named `own_traj`) contains modified dynamically feasible trajectories for each trajectory `own_traj_orig`. Specifically, for each flight plan, additional waypoints are added such that transitions between the original flight plan waypoints comply with typical aviation restrictions (e.g., no more than 30°angle-of-bank, etc.) when undertaking climbing flight, acceleration, or early turn between waypoint line segments.

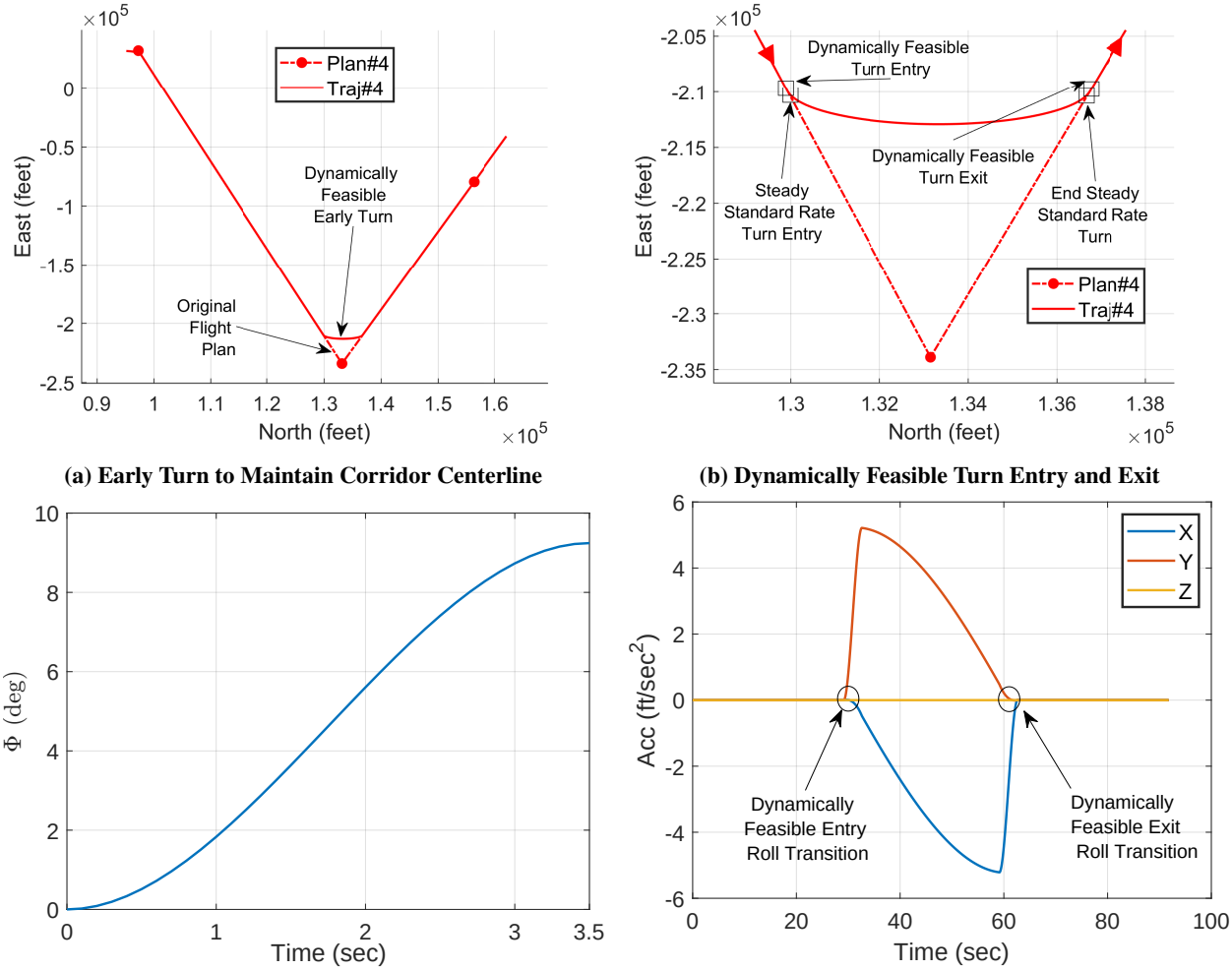
Each trajectory is a complete flight including hover take-off, performing a basic prescribed terminal area departure, a climb-cruise-descent, and a prescribed terminal area arrival procedure. Multiple trajectory parameters, such as rate of climb, airspeed, and altitudes, are randomly (uniformly) distributed across the 3000 trajectories. The frame of reference used is the Earth-fixed North, East, and Down inertial frame. The waypoint data format is that of a third-order Bernstein polynomial. So for each waypoint, there are x-, y-, and z-axis positions, velocities, and accelerations, as well as time. For the own_traj_orig flight plans, all accelerations are zero, which makes these waypoints equivalent to traditional flight plans. The dynamically feasible own-ship trajectories add additional transition waypoints (with non-zero accelerations) to ensure that the entire complete trajectory is smooth (continuous) in velocities and accelerations. Figure 2 depicts sample cases of the original flight plans (labeled Plan) and the corresponding dynamically feasible trajectories (labeled Traj) found in data set one. As shown in the figures, these runs consist of standard departures and arrivals combined with random climb, cruise, and descent legs (as noted in Fig. 2). Uniform variation is applied to velocities, headings, and climb and descent rates.



(a) Top View (North and East) **(b) Side View (North and Altitude)**

Fig. 2 Data Set 1: Flight Plans and Dynamically Feasible Trajectories

Some further explanation is warranted on what converting the nominal flight plans of data set one into the corresponding dynamically feasible trajectories entailed. In the North and East plane, the process included computing early turn points to exactly transition from the centerline of one linear segment to the centerline of the follow on segment. This process is typically desired for flight in airspace corridors. An example of this process is shown in Fig. 3a. This process of maintaining centerline between segments was then augmented with a dynamically feasible roll rate entry (exit) turn that satisfied maximum angle-of-bank, turn-rate, and acceleration limits. These dynamically feasible entry/exit turns were also applied to early turns to ensure dynamically feasible early turns were performed between linear segments. An example of the additional waypoints for dynamically feasible early turn entry is shown in Fig. 3b. While it does not appear to be (due to axis scaling), the turn segment between the dynamically feasible entry points is a circular arc.



(a) Early Turn to Maintain Corridor Centerline

(b) Dynamically Feasible Turn Entry and Exit

(c) Example Entry Roll Angle vs Time

(d) Example Transition Heading Frame Accelerations

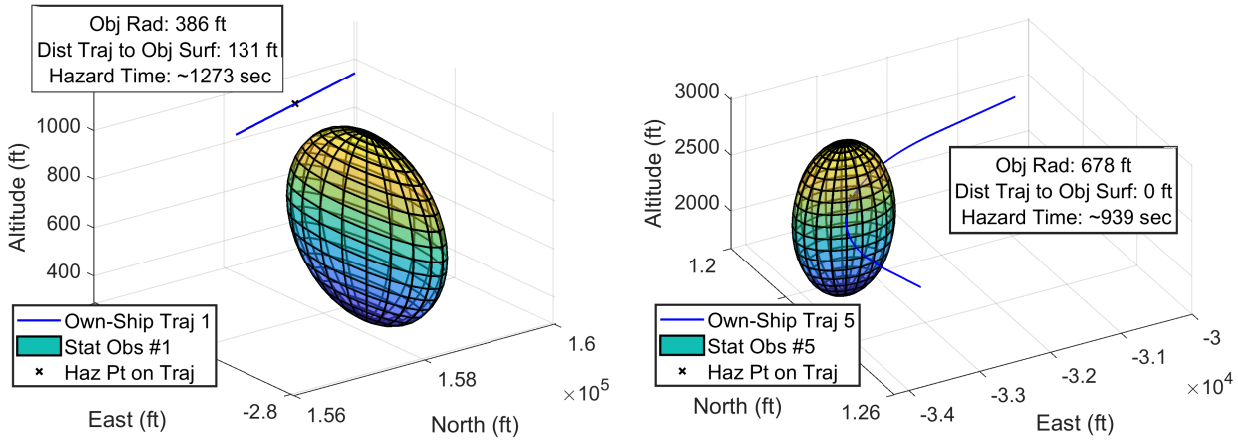
Fig. 3 Data Set 1: Flight Plans and Dynamically Feasible Trajectories

The feasible entry turns are performed using a smooth roll angle transition from wings level to the steady state angle for the desired turn rate (e.g., standard rate turn). An example roll angle transition is shown in Fig. 3c. Use of the roll transition profiles, in conjunction with the early turn entry/exit points ensures smooth continuous trajectories that comply with typical aerospace limitations (e.g., turn rate, angle-of-bank, and acceleration). This turning process of entry/exit transitions together with steady-state turns results in a smooth acceleration profile with an example shown in Fig. 3d.

The second data set contains 3000 examples of stationary obstacles. Each of these obstacles was randomly created to be used with the corresponding own-ship dynamically feasible trajectory. In this manner, a user that selects the tenth own-ship trajectory is assured that the tenth stationary obstacle is guaranteed to present a collision hazard to that trajectory. As all the trajectories takeoff and land at identical locations, then several other obstacles (other than the

one with the same trajectory index) are also likely to present a collision hazard to the selected own-ship trajectory run. Users can then create sample problems with multiple stationary obstacles by selecting one own-ship trajectory, the corresponding stationary obstacle, and one or more of the other stationary obstacles that are in close proximity to the chosen own-ship trajectory.

The creation of the stationary obstacles was performed by the script `Generate_Stat_Obs.m` and the results are stored in the `Data_Set_2.mat` file. The format for this data is found in the generating script `Generate_Stat_Obst.m`. For a given own-ship trajectory, uniform variation was performed on the hazard time or the time during the trajectory at which the hazard is closest to the selected trajectory. Once the hazard time was determined, then uniform variation was performed to place a randomly sized sphere's center somewhere in the plane perpendicular to the trajectory with a random distance from the trajectory position at the hazard time. Data set 2 contains the following obstacle data for each run: obstacle center position (3-vector), object radius (scalar), minimum perpendicular distance of surface of the sphere to trajectory (scalar), hazard time (scalar), position on trajectory at hazard time (3-vector), and the corresponding trajectory number (scalar). An example of two of the randomly created stationary obstacles are shown in Fig. 4. Note that the obstacles are arbitrarily sized and located, and they occur at random times in the own-ship trajectory. Moreover, some obstacles may intersect the own-ship trajectory, and others are just in close proximity.



(a) Stationary Obstacle for Trajectory #1

(b) Stationary Obstacle for Trajectory #5

Fig. 4 Data Set 2: Example Stationary Obstacles

The implementation of the moving obstacles is similar to that of the stationary obstacles. The moving obstacle data which found in `Data_Set_3.mat` was produced by the script `Generate_Mov_Obs.m`. The obstacle itself is randomly created, and associated data is stored in the same manner as the stationary obstacle. In addition to the random data for the spherical obstacle at hazard time, an additional linear trajectory for the obstacle is created that ensures the obstacle reaches the correct position at the prescribed hazard time. This trajectory has random speed, both random North and East velocity components, and a random vertical velocity component. The resulting linear trajectory is stored as a Bernstein polynomial. Figure 5 shows a typical example (trajectory #5) of both own-ship and moving obstacle trajectories. As with the stationary obstacles, the moving obstacle cases are correlated to ensure a hazard to collision with the corresponding own-ship trajectory. For example, trajectory number 135 of both own-ship and moving obstacle trajectories is guaranteed to present a near collision scenario. However, as in the static obstacle situation, one or more of the 3000 runs may also present additional hazard collision scenarios.

The control effector scenarios make up the last of the challenge problem data sets. The script `Generate_Failures.m` was used to create the effector failures found in `Data_Set_4.mat`. The GUAM simulation facilitates a wide array of both aerodynamic and propulsive effector failures. However, for the challenge problems, only four types of effector failures were utilized. These include hold last (i.e., stuck effector), scaling either before or after effector dynamics (e.g., reduced output), and control reversal (applicable to aerodynamic effectors only). The failure scenarios occur at a uniformly random time and for a random duration (e.g., up to 100 seconds) within each own-ship trajectory. The effector numbering scheme for the NASA Lift+Cruise vehicle is shown in Fig. 6.

In this section, we provided greater detail on the data sets available to support autonomy challenge problems. As

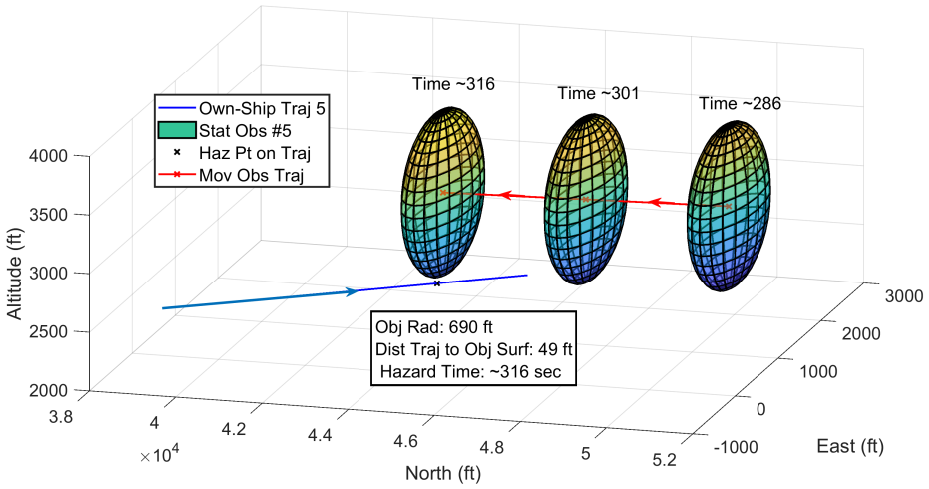


Fig. 5 Data Set 3: Example Moving Obstacle Trajectory

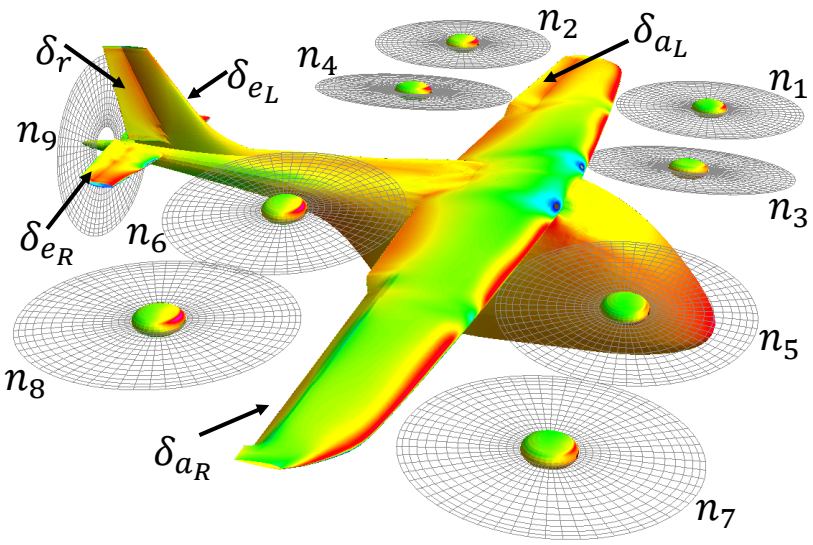


Fig. 6 NASA Lift+Cruise Effector Numbering

outlined previously, these data sets facilitate the composition of increasingly difficult autonomous tasks. While GUAM v1.1 provided the precursor data required, to date our research team has not released specific autonomy challenges. This lack of specific autonomy tasks was intentional due to the lack of some necessary simulation capabilities in GUAM. As will be discussed in section V, the next simulation release GUAM v2.0 will include high-fidelity visualization (i.e., Unreal Engine v5), a preliminary cognitive architecture, and enhanced ability to incorporate other software (e.g., Python®, ROS2). GUAM users can expect specific autonomy challenges with that simulation release to include traffic pattern entry at non-controlled airfields and obstacle collision avoidance.

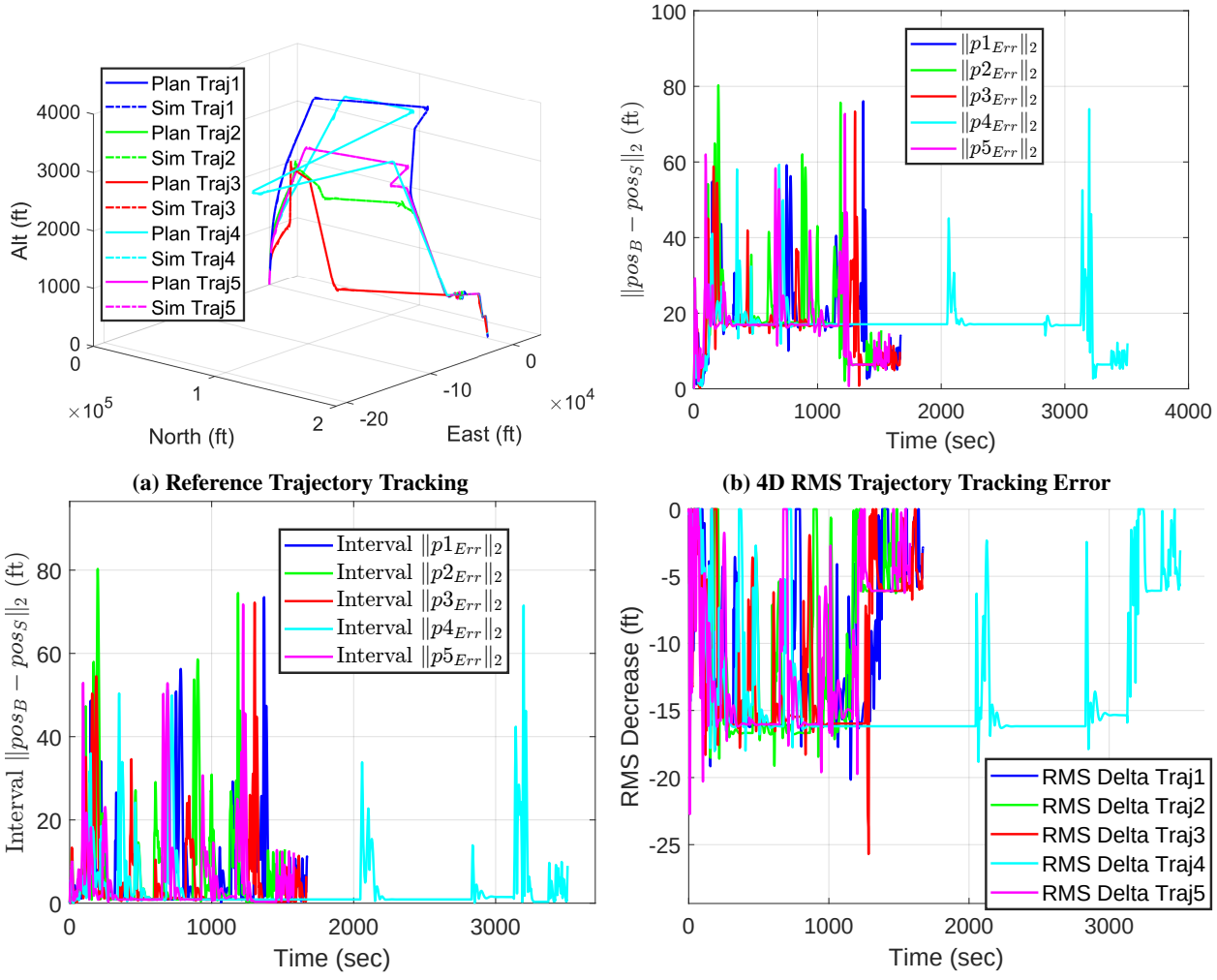
IV. Experimental Results

This section describes some basic experiments conducted to demonstrate the use of challenge problem data sets in conjunction with the GUAM v1.1 simulation. Additionally, some basic performance metrics are presented for evaluating autonomous performance for various contingencies experienced during normal flight operations.

One of the most basic autonomy tasks in aviation is the capability to fly a given flight trajectory. To this effect,

we flew a subset of the own-ship dynamically feasible Bernstein polynomials trajectories available in the autonomy challenge problem `Data_Set_1.mat`. These flights were accomplished using the existing unified Integral Linear Quadratic Regulator (iLQR) baseline controller and the polynomial aero-propulsive model, which are both supplied with GUAM. Figure 7 shows the results of these experiments.

Figure 7 shows the spatial own-ship trajectories 1-5 and the corresponding as-flown GUAM simulation results. A simple autonomy analysis question is whether or not this tracking performance is adequate. The most basic metric for tracking performance is the root mean square (RMS) error between the desired flight trajectory and the as-flown trajectory at each given time step. This temporally constrained RMS error metric is shown in Fig. 7b. While the metric is shown at each instant for all 5 flights, basic statistical performance metrics are readily available such as mean 4D RMS error and the associated standard deviation across individual flights as well as across a complete set of flights. In most flight regimes, the primary objective is to track the desired trajectory spatially, but a fairly large amount of temporal flexibility exists. For instance, when tracking an air corridor, it is important to stay within the corridor, but being at the planned location along the corridor at a specific time is not critical. For this reason, a second tracking



(a) Reference Trajectory Tracking

(b) 4D RMS Trajectory Tracking Error

(c) 3D RMS Trajectory Tracking Error

(d) Impact of No Temporal Tracking Requirement

Fig. 7 Trajectory Tracking Performance

metric is demonstrated in Fig. 7c. Here the same RMS of desired and actual position is computed, but instead the minimum RMS error between the desired position on the prescribed trajectory is found at the corresponding as-flown trajectory but for a range of times (e.g., $t_{des} \pm 2$ seconds). Figure 7d shows the improvement in measured tracking performance achieved simply by recognizing what is of primary importance to measure.

The goals of creating and defining performance metrics for autonomy are three fold. First they should measure what

is important during a particular phase of flight. For example, the metric for tracking performance during precision landing will emphasize altitude much more that during cruise within a prescribed flight corridor whose only altitude requirement is ±300 feet. The RMS spikes seen in Fig. 7b can be shown to correlate with various step maneuvers (e.g., heading change, altitude change etc.) Each of these phases of flight vary in what should be emphasized when measuring performance. The second function of performance metrics is to enable a quantitative means of comparing performance between controllers, algorithms, etc. This is important for design within a research group, but by using a common simulation framework and identical data sets, it also facilitates comparison among research groups. Lastly performance metrics are likely to be used in real-time or near real-time by autonomy algorithms as an aid in cognitive decision processing.

A second set of demonstration experiments was conducted using the challenge problem data sets. In particular, our research group's previous work combing Bernstein polynomials with the robotics algorithm optimal reciprocal collision avoidance was applied to the moving obstacle problems found in data set three [11]. The results are shown in Fig. 8. Figure 8a shows the top view of a collision avoidance maneuver for case 1 of the challenge problem own-ship and the moving obstacle data sets. In this figure, the blue trajectory is the baseline dynamically feasible trajectory, and the red trajectory is an avoidance maneuver to the right. The yellow trajectory is the moving obstacle with the obstacle sphere shown at the time of closest approach to the nominal own-ship trajectory. Figure 8b shows the same avoidance maneuver from a three-dimensional perspective. Figure 8c, shows the smooth (feasible) nature of the avoidance maneuver. While their are many ways to measure collision avoidance performance, the most basic is to ensure that the minimum prescribed separation distance is maintained, as seen in Fig. 8d. Collision avoidance in three-dimensional space can occur in many ways (turns, altitude, etc.), a basic metric question to ask is what is the best way to avoid an obstacle. Figure 8e shows a left-hand avoidance maneuver for the same collision scenario shown previously. A potential measure of a successful collision avoidance maneuver (i.e., one which preserves desired minimum separation distance) is to determine how aggressive was the maneuver required to achieve the minimum separation. In this way, Fig. 8f shows the Euclidean distance between the original own-ship trajectory at each time instant for each of the avoidance maneuvers (Left and Right). Examining the deviation between the avoidance maneuver and the original trajectory could be used as a cognitive metric to enable an autonomous algorithm to select among many successful choices of potential avoidance trajectories. In this way, in Fig. 8f, the right avoidance maneuver could be viewed as easier to perform.

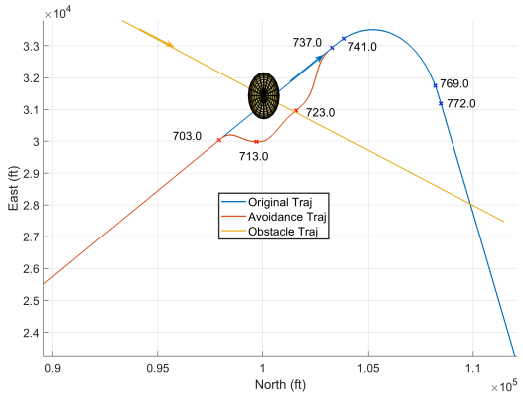
The experiments shown in this section were intended to familiarize GUAM users with the available data sets and to show representative autonomous capabilities and some basics metrics that could be used to evaluate performance. With some basic understanding of the simulation and available data sets, users can readily examine complex autonomy scenarios.

V. Discussion and Future Work

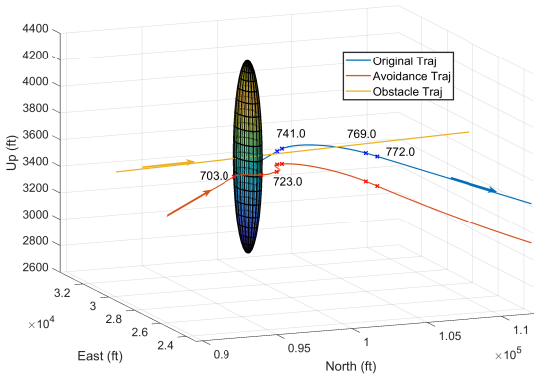
This research presented a relevant UAM vehicle (NASA Lift+Cruise) simulation architecture that would foster the autonomy research community to collaborate and compare performance in an open-source environment. To enable these goals, we provided an open source simulation architecture and establish the tools necessary for autonomy challenge problems. The initial version of GUAM v1.1 has many of the features necessary to begin tackling the research challenges associated with autonomous flight in the context of emerging AAM sector. However, some important features for other autonomy research fields are not provided with this initial simulation release. In particular, perception is not readily supported in this release (e.g., no high-fidelity visualization). Additionally, the field of autonomous cognition (decision making) is not implemented. In order to support these fields (and others), GUAM v2.0 is pending NASA software release approval. This updated simulation will support high-fidelity visualization, provide a basic cognitive architecture, provide simplified ROS2 interface (no Matlab® toolbox required) via s-functions, and support external Python® algorithms. These features will facilitate cognitive algorithms outside the simulation framework and provide the machine learning/computer science communities access to a relevant autonomy simulation environment. In addition to a GUAM update to v2.0, a NASA Technical Memorandum that documents GUAM in detail is under publication review.

The GUAM repository can be found on github at `https://github.com/nasa/Generic-Urban-Air-Mobility-GUAM`. More details on the autonomy community challenge framework are located on the NASA TTT Revolutionary Aviation Mobility (RAM) site `https://nari.arc.nasa.gov/ttt-ram/community`.

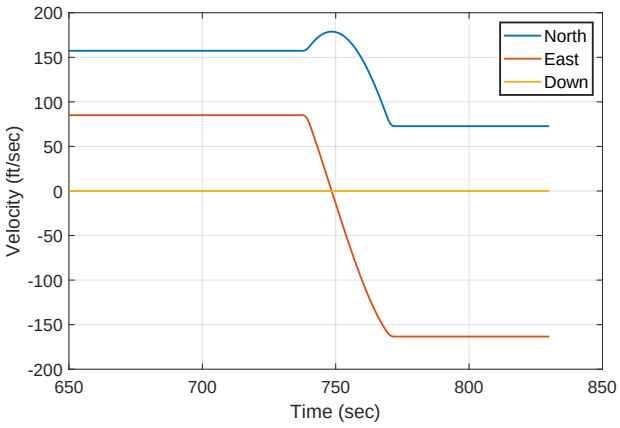
For researchers who utilize the GUAM simulation, please cite the GUAM github repository (see citation below). This repository, and associated variant forks are intended to be an autonomous flight hub to assist those trying to identify, collaborate, and compare with relevant research teams and their work.



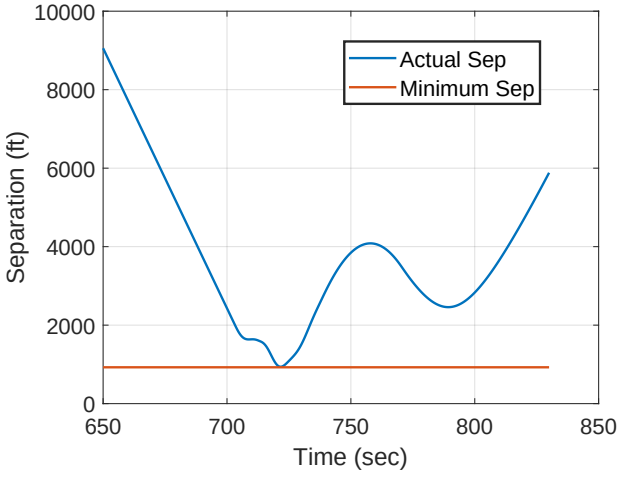
(a) Avoidance Maneuver Right (Top View)



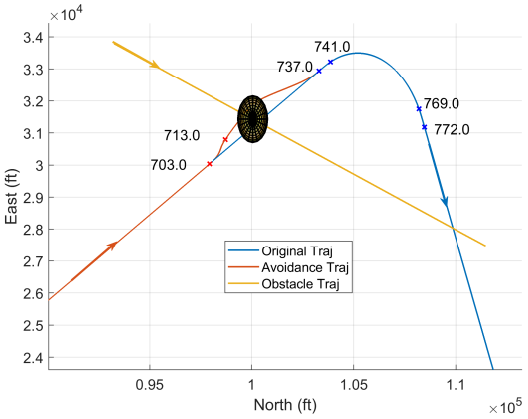
(b) Avoidance Maneuver Right



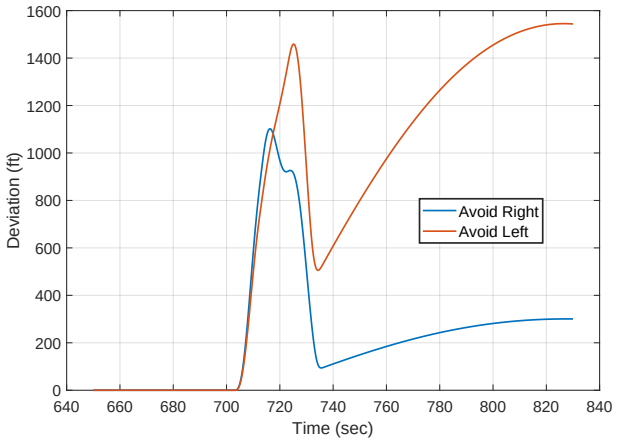
(c) Maneuver Right Velocity



(d) Maneuver Separation Distance



(e) Avoidance Maneuver Left (Top View)



(f) Maneuver Deviation

Fig. 8 Collision Avoidance Performance

Citation: Acheson, M. J., Cook, J. W., Simmons, B. M., et al., Generic Urban Air Mobility (GUAM) Simulation v1.1, Released 21 May, 2024, https://github.com/nasa/Generic-Urban-Air-Mobility-GUAM.

Acknowledgments

This research was supported by the NASA Aeronautics Research Mission Directorate (ARMD), Transformative Tools and Technologies project, under the Autonomous Systems / Intelligent Contingency Management subproject. Additional thanks to outstanding researchers who contributed to the GUAM simulation including: Jacob Cook, Benjamin Simmons, Steve Derry, and Bart Bacon with NASA and Tom Britton with the Science and Technology Corporation.

References

[1] Wikipedia contributors, "Boeing MQ-25 Stingray — Wikipedia, The Free Encyclopedia," , 2024. URL https://en.wikipedia.org/w/index.php?title=Boeing_MQ-25_Stingray&oldid=1256290756, [Online; accessed 19-November-2024].

[2] Wikipedia contributors, "Boeing X-45 — Wikipedia, The Free Encyclopedia," , 2024. URL https://en.wikipedia.org/w/index.php?title=Boeing_X-45&oldid=1256292771, [Online; accessed 19-November-2024].

[3] Wikipedia contributors, "Instrument landing system — Wikipedia, The Free Encyclopedia," , 2024. URL https://en.wikipedia.org/w/index.php?title=Instrument_landing_system&oldid=1244653405, [Online; accessed 19-November-2024].

[4] Shappell, S., Detwiler, C., Holcomb, K., Hackworth, C., and Boquet, A., "Human Error and Commercial Aviation Accidents: A Comprehensive, Fine-Grained Analysis Using HFACS," Tech. rep., 2006. Embry-Riddle Aeronautical University.

[5] Shappell, S., and Wiegmann, D., "HFACC Analysis of Military and Civilian Aviation Accidents: A North American Comparison," *ISASI Proceedings*, Vol. 8, 2004.

[6] NTSB, "Accident Report: NTSB/AAR-10/03 PB2010-910403," Tech. rep., 2010. Federal Aviation Administration.

[7] NTSB, "Accident Report: NTSB/AAR-90/06 PB90-910406," Tech. rep., 1989. Federal Aviation Administration.

[8] Holbrook, J., "Exploring methods to collect and analyze data on human contributions to aviation safety," *49th International Symposium on Aviation Psychology*, 2021, p. 110.

[9] Holbrook, J., Prinzel III, L. J., Stewart, M. J., and Kiggins, D., "How Do Pilots and Controllers Manage Routine Contingencies During RNAV Arrivals?" *Advances in Safety Management and Human Performance: Proceedings of the AHFE 2020 Virtual Conferences on Safety Management and Human Factors, and Human Error, Reliability, Resilience, and Performance, July 16-20, 2020, USA*, Springer, 2020, pp. 331–338.

[10] Cook, J. W., and Gregory, I. M., "A Robust Uniform Control Approach for VTOL Aircraft," Vertical Flight Society – 2021 Autonomous VTOL Technical Meeting and Electric VTOL Symposium, Virtual, January 27, 2021.

[11] Houghton, M. D., Acheson, M. J., Patterson, A. P., Oshin, A., and Gregory, I. M., "Combined Bernstein Polynomial Collision Avoidance Differential Dynamic Programming for Trajectory Replanning and Collision Avoidance for UAM Vehicles," *AIAA SciTech 2023 Forum*, 2023. URL https://arc.aiaa.org/doi/abs/10.2514/6.2023-2544.