Flight Test Evaluation of Automation-Induced Oscillations

David Zahn[1]
NASA Ames Research Center, Mountain View, California, 94035, U.S.A

Gayle Patterson[2] and Paul Sampson[3]
American Systems Corporation, Chantilly, Virginia, 20151, U.S.A.

Saravanakumaar Ramia[4], Wayne Ringelberg[5], and Adam Yingling[6]
NASA Armstrong Flight Research Center, Edwards, California, 93524, U.S.A

Sarah Eggum[7]
Flight Research Aerospace, Mountain View, California, 94043, U.S.A.

Flight-test systems analysis techniques are applied to the case of aircraft automation-induced oscillations (AIO). Emerging technological advancements in flight control automation and artificial intelligence have created a need for evaluating the age-old aerial phenomenon of undesired aircraft oscillations. This paper is an overview on the study of AIO as interactions between human pilot, software engineer, and vehicle dynamics (e.g., latency) with the presence of winds aloft. Flight-test evaluation included a broad, non-comprehensive theory on cyclical and non-cyclical oscillations and a preliminary evaluation of the ecosystem from software engineer, airspace architect, and human test pilot.

I. Introduction

New forms of highly automated Advanced Air Mobility (AAM) aircraft, such as electric vertical takeoff and landing (eVTOL) vehicles, could transform transportation, cargo delivery, research, and a variety of public services. The National Aeronautics and Space Administration (NASA) conducted a series of demonstrations in collaboration with the Defense Advanced Research Projects Agency (DARPA) and Sikorsky Aircraft (Lockheed Martin, Bethesda, Maryland) to progressively evaluate autonomous technologies.

The Sikorsky experimental S-70 Black Hawk, an Optionally Piloted Vehicle (OPV), was utilized as an intruder aircraft during flight tests where the S-76B rotorcraft, Sikorsky Autonomous Research Aircraft (SARA), served as the "ownship", to fulfill the role of a surrogate AAM vehicle. Research included Flight Path Management (FPM) algorithms for automated pilot-selected alternatives to potential converging trajectories with simulated traffic and Hazard Perception Analysis (HPA) for human autonomy studies with Aircraft Collision Avoidance Systems for Rotorcraft (ACAS XR). Human-to-machine and machine-to-machine interfaces and interactions, flight-path algorithms, as well as middleware (MW) software for the automated execution of maneuvers were evaluated during

---

[1] Research Pilot, ARC-AFO, AIAA member.
[2] AST Theoretical Simulation Techniques, AFRC-540, AIAA nonmember.
[3] AST Theoretical Simulation Techniques, AFRC-540, AIAA nonmember.
[4] AST Nav Guidance & Control Systems, AFRC-540, AIAA nonmember.
[5] Chief Test Pilot, AFRC-410, AIAA member.
[6] Technical Lead, AFRC-530, AIAA member.
[7] Program Metrics Analyst, ARC-AFS, AIAA nonmember.

flight maneuvers in a relevant flight environment. Specifically, four-dimensional trajectories (4DT) (latitude, longitude, altitude, and time) were also evaluated.

Legacy flight tests have long evaluated oscillations due to pilot inputs known as pilot-induced oscillation (PIO). The PIO phenomenon are the unintended sustained or uncontrollable undulations and jittering, which occur when the pilot attempts to compensate for the aircraft reaction to a previous pilot input, which is an overcorrection in the opposite direction [1]. An oscillation may occur as the result of a reduced phase margin induced by the lag of a pilot's response or in a lag of the response rate of flight instrumentation. For example, when a car driver experiences a rapid loss in car tire pressure, a common reaction is to overcontrol the steering wheel and chase equilibrium, which could result in the total loss of vehicle control. Similarly, the PIO of an aircraft can result in upward, downward, or lateral "porpoising" attributed to the pilot input frequency against the aircraft control frequency. The PIO is evaluated as a handling quality factor against an established assessment scale [1].

The two types of automation-induced oscillations (AIO) experienced during the flight test were a lateral "ratcheting" characterized by a one 1-Hz lateral oscillation that seemed most prevalent during climbs and descents with a turn. This condition provided an example for future flight-test evaluations of a cyclical AIO that can be identified and solved through software analysis. The second type of AIO identified during the flight test was in the pitch axis that occurred intermittently in which the aircraft automation attempted to command 40 kts to 144 kts groundspeed targets in three- to six-s intervals. As the targeted airspeeds were in groundspeed instead of indicated or true airspeed, variations in wind azimuths relative to the aircraft heading resulted in lower indicated airspeed targets below the S-70 turn coordination speed. The two oscillation types were experienced and evaluated by all test pilots and remediated through discussions with the software engineering team. This paper provides an outline for the discussions, roles, and responsibilities of the software engineer and future test pilots evaluating increasing levels of automation and artificial intelligence (AI) with new and novel aircraft designs, propulsion systems, and flight profiles. Throughout the flight demonstrations, pilots experienced oscillations in the aircraft for certain maneuvers under certain conditions of the autonomy modes. Data corroborated the phenomena. This flight-test report provides a description, analysis, and findings for the AIO that occurred during the flight tests in Stratford, Connecticut August 2022, November 2022, February 2023, May/June 2023, and October 2023.

II. Test Description

This section outlines a description of the multiple aircraft under test. This section also provides discussion of the NASA-designed architecture and configuration and presents a brief description of the test site.

A. Aircraft Description

The NASA Integration of Automated Systems (IAS) test program utilized two surrogate autonomous-capable aircraft, which hosted NASA research algorithms to develop software to maintain a low flight safety risk. Sikorsky Innovations (the Advanced Concepts group within Sikorsky), located in Stratford, Connecticut provided two research aircraft: the Sikorsky Autonomy Research Aircraft (SARA) S-76B and the Optionally Piloted Vehicle (OPV) S-70 Black Hawk TM helicopter as IAS research testbeds. Both helicopter test beds incorporate the Sikorsky software packages MATRIXTM and the DARPA-Sikorsky Aircrew Labor In-cockpit Automation System (ALIAS) systems. The IAS team identified and provided key technologies needed to evolve Urban Air Mobility (UAM)/AAM into progressively more complex, automated operations utilizing these two testbeds and their associated systems. Interfacing with the MATRIXTM system was achieved through a Sikorsky-developed software called the Autonomy Mission Manager (AMM), which served as the interface point for external commands to be sent to the Vehicle Management Computer (VMC) and for state data to be made available[2]. The aircraft were operated by a NASA research pilot and a Sikorsky safety pilot.

B. Software Description

The MW software architecture is based on the Expandable Variable-Autonomy Architecture (EVAA), which is an implementation of a Multi-Monitor Run-Time Assurance (MM-RTA) architecture that provides a robust methodology for evaluating unmanned and autonomous systems. The EVAA framework coordinates various functionalities with risk-based logic, safely bounding untrusted behavior and relieving the requirement to certify all guidance logic possibilities. The framework is structured to allow the addition and removal of monitors, sensors, and aircraft models with a minimum of validation and verification testing.

The MW software architecture is shown in Fig. 1 along with the IAS-specific modules. The Flight Executive and Scheduler are the main components of the MW, loading and running the core modules and plug-ins. The plug-ins are custom components of each project and consist of sensor couplers, command couplers, and Flight Test Services.

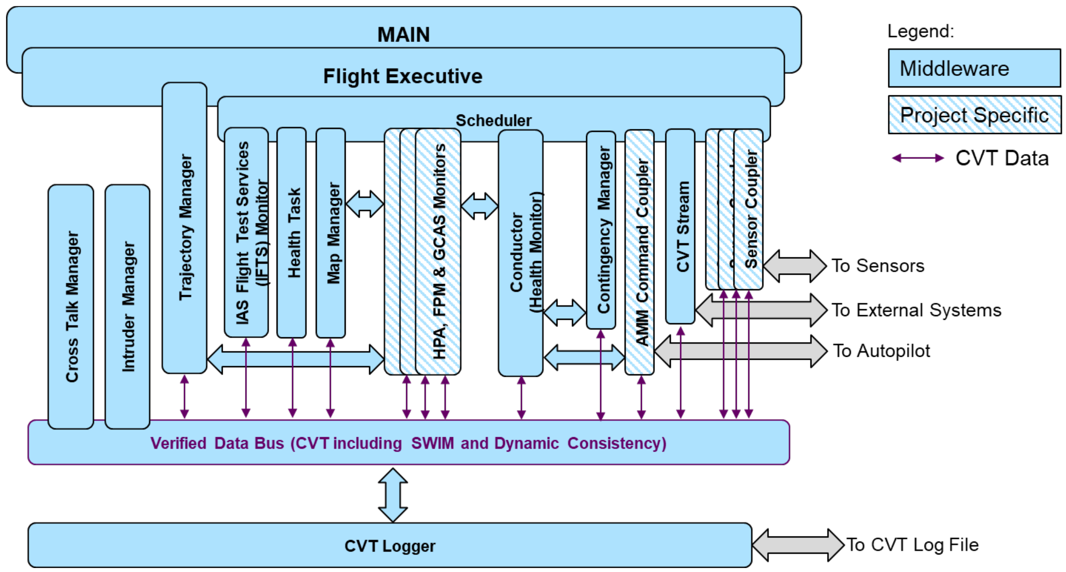

Fig. 1 Details of the software architecture of the NASA Integration of Automated Systems middleware.

Sensor couplers provide an interface with aircraft sensors, storing the data in the Current Value Table (CVT), a data dictionary accessible to all MW components. The CVT data are telemetered to external systems via the Pub/Sub CVT Stream. A single command coupler provides the ability to send commands to the aircraft. A command coupler may also be a sensor coupler. The couplers isolate the effects of changes to the interfaces and enable the MW to provide a stable data platform for the hosted applications, namely the technologies under test.

Each safety monitor uses the data in the CVT to detect potential threats and issues a command to handle the highest priority threat. The last flight-test configuration included four monitors, but only one monitor was active at a time. The MW sends commands from the selected monitor to the aircraft via the command coupler. The MW consists of:

1) Middleware Core
2) Sikorsky Autonomous Research Aircraft (SARA) Autonomous Mission Manager (AMM) Command Coupler
3) IAS Flight Test Services (IFTS) monitor (including support for Autoland)
4) Sensor Couplers
    a. WindCalc Coupler,
    b. Ping Coupler, and
    c. Ping Station Simulator (PingSim) Coupler.
5) Monitors
    a. Hazard Perception and Avoidance (HPA),
    b. Flight Path Management (FPM), and
    c. Improved Ground-Collision Avoidance System (iGCAS).

The AMM coupler is both a sensor and command coupler, storing aircraft state data in the CVT and sending trajectory, takeoff, and land commands to the aircraft. The AMM Module is a component added to AMM providing a MW interface to the SARA and OPV aircraft. The IFTS monitor receives commands from the MW engineer and the aircraft pilot, forwarding the commands to the AMM Coupler for transmission to the aircraft. Wind Calculation (WindCalc) is a sensor coupler plug-in which estimates the wind speed and direction, using data provided by multiple sensors on the aircraft The Ping Coupler pulls ADS-B tracks from the PingStation and forwards any threats to the Intruder Manager (Fig. 2).

In the second round of tests, the traffic on the data link was reduced by adding a CVT Gateway (GW) computer in the ground station to collect the CVT Stream data and forward it to the ground clients. The FPM Autonomous Operations Planner (AOP) was added as a new monitor. Additional support was also added to support flying two aircraft, one as “ownship” and one as “intruder,” to exercise the traffic deconfliction abilities of the two algorithms. Both aircraft used the MW with slightly different configurations. For the third round of tests, support was added for

HPA: iGCAS and auto-land capabilities. For more information, refer to "Integration of Automated Systems Test Campaign". [3]

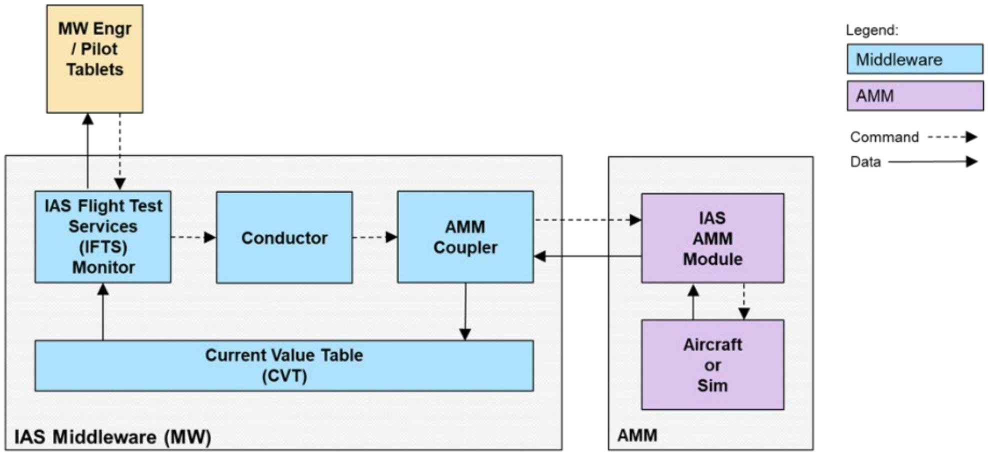

Fig. 2 The NASA Middleware command and data configuration.

C. Test Site Description

Flight crews consisted of four Sikorsky safety pilots and three NASA research pilots that were outfitted with biometric kits which included eye tracking, brain activity monitoring, heart rate, body temperature, and respiratory data. All flights were launched and recovered to and from class D airspace at Bridgeport Airport (KBDR). The area of operation for the flight test was 10 NM x 24 NM southwest of the airfield over the Long Island Sound from 1500 ft to 7000 ft mean sea level (MSL). All flights were also conducted over unpopulated areas above the water in daytime visual meteorological condition (VMC) as seen in Fig. 3.

Fig. 3 Map of operational boundaries at KBDR and Long Island Sound.

The ground control station (GCS) was in a trailer outside of the Sikorsky hangar at the launch facility (KBDR) airport. Communication and control, as well as telemetry, data were sent to and from ground control with both research aircraft. The class D airport tower was used while flights were conducted in controlled airspace. When not in controlled airspace, advisories were called out from the Sikorsky radar facility located just north of the airspace at the Sikorsky private heliport (KJSD).

III. Background

This section provides a brief description of automation, oscillation, and the human-machine interface that occurs with these events. This section is a high-level summary of the impact of automation on these events.

A. Definition of Automation

"Automation" is defined in this paper as a process performed automatically using software that had traditionally required human cognition to complete. Although similar to existing autopilots in some ways, the automation logic under test utilized complex deconfliction algorithms and real-time external data sources to generate high-level aircraft 4DTs. These time-based mathematical expressions were continuous and fully differentiable aircraft state descriptions across the planned flight profile. The 4DTs were then rasterized into discrete velocity-based state information before being sent over to the Autonomy Mission Manager (AMM) for interpretation into aircraft primitive commands such as power, pitch, roll, yaw, and heave commands [3].

B. Definition of Oscillation

"Oscillation" is defined in this paper as the periodic motion of the aircraft about one or more axes. These oscillations typically occurred in the pitch and roll axes, were limited in amplitude, and displayed intermittent dampening. Sometimes referred to as Limit-Cycle Oscillations (LCO), this type of oscillation emerges in nonlinear systems and is characterized by a constant amplitude and frequency for as long as the conditions exciting the system remain unchanged. Refer to Fig. 4 "cyclical stable" [1].

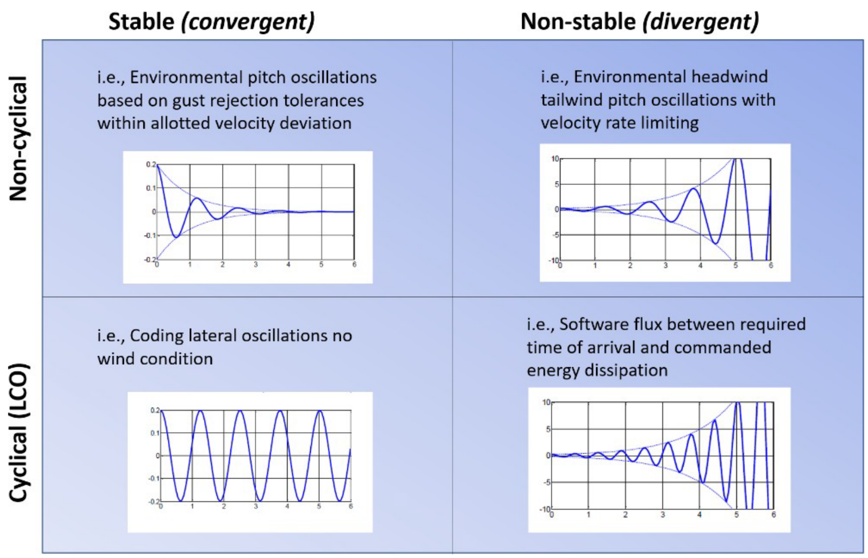

Fig. 4 Categorization matrix for automation-induced oscillations.

C. Overview of Pilot-Induced Oscillation

Pilot-induced oscillations, sometimes referred to as pilot-involved oscillations, or as unfavorable aircraft-pilot couplings (APC), are unintended excursions in aircraft attitude and flight path caused by anomalous interactions between the pilot and the aircraft experienced as sustained oscillatory motions [4]. The presumption of blame implicit in the term "pilot-induced" in APC events are deficiencies in the design of the aircraft, displays, and the flight control system that produce adverse pilot coupling [5]. These result from efforts when the pilot inadvertently commands an often-increasing series of corrections in opposite directions; each one is an attempt to control the reaction of the aircraft to the previous input with an overcorrection in the opposite direction.

D. Identification of Automation-Induced Oscillation

Automation-induced oscillations are unfavorable, periodic, unintended excursions in aircraft control about one or more axes due to commands from an autonomous, AI or machine learning (ML) system. These outer-loop systems are unrelated to an autopilot, fly-by-wire, or stability augmentation systems that have been in use for more than sixty years. The emergence of AIO is present when airspace management, route planning, or procedure design logic are dynamically generated to compute flight maneuvers in the departure, enroute, and landing phases of flight. Specific attention should be given when an increasingly autonomous airspace management system interfaces with a highly augmented or completely automated flight system. Automation-induced oscillations are the unintended consequence of mandating required times of arrival of multiple aircraft with a continuous conversion from time-based to discrete velocity-based commands. Automation-induced oscillations can result from one or more autonomy system interfacing to generate flight profiles. The single autonomy system represents one autonomous system route generator controlling aircraft trajectories in a single loop. The dual autonomy systems are represented by wo independent autonomous systems incorporating aircraft trajectories in real time (NASA MW and Sikorsky AMM) by way of a route generator, autonomy kit and autonomous system controlling the aircraft.

IV. Findings

There are 64 events identified during the flight tests on both SARA and OPV (Table A-1 in the appendix). An AIO event is identified when AMM Time Error exceeded +/- 1 s oscillating across both thresholds during the event. At such events, the pitch oscillations exceeded +/- 2 deg and the AMM Target Velocity was also oscillating. The mean duration of the AIO events is nearly 4.4 min.

For simplification, AIO are defined by axis, cyclical versus non-cyclical, and stable versus divergent oscillatory motion. The categorization matrix below was established to determine the origin of the oscillation and support communication between the aircrew and the software design teams during the flight test (Fig. 4). Although there are many possible variations of AIOs when integrating AI/ML into flight path trajectories, the lateral and pitch oscillations were the only adverse behaviors observed for this report. The following summaries will explore the origin of the oscillator (the destabilizing forces introduced through the MW commands) and whether it was caused from software coding or a response to an environmental condition. To define the AIO:

1) Measure the frequency (number of overshoots).
2) Determine if a dampening ratio (decay function over frequency) was present.
3) Determine if the oscillations are neutrally stable, divergent or convergent.

A. Lateral Automation-Induced Oscillations

The first AIO encounter type was in the lateral axis in which a 1-Hz "ratcheting" was observed when the NASA MW was given trajectory commands to SARA. The oscillation was observed from the flight deck as well as from the GCS and simulator.

1. One-Hertz ratcheting – Cyclical Automation-induced Oscillations

<u>Flight Deck Observations:</u> Researchers observed intermittent neutrally stable oscillation with zero dampening from AMM commands independent of head-, cross-, or tailwind components. The origin of the oscillation was based on software inputs and was not observed during any manually controlled flight. After discussion with ground control and the autonomy team, it was discovered the automation logic placed waypoint targets at 3-ft intervals in front of the aircraft. These truncated commands resulted in a 1-Hz neutrally stable lateral oscillation when climbing and turning. The resulting motion was experienced in the aircraft with a 1-Hz lateral "head bobbing" pattern during the maneuver. When reviewing the data, it was determined to be a software problem based on the "cyclical" nature of the oscillation. The velocity controller adjusted the 4DT continuous description of position in time to rasterize deltas in time or position error. The PID controller "smoothing" changed the interpolation between the velocity and time stamp targets (required time of arrival) for reduced time error in waypoint executions of commanded versus actual trajectories. The cyclical-neutrally stable AIO in the roll was an indirect consequence of the poor turn trajectory provided. The anomaly was resolved by redefining the number of trajectory points.

<u>Ground Control Observations:</u> The automation algorithm worked in areas where the MW algorithms created closely spaced points, such as in a turn, as the target velocities between adjacent points would differ minimally. In long straightaways, however, such as transit to the setup area, the algorithms spaced points widely. The entire trajectory needed to be telemetered to the ground and stored in log files, and large lists of points quickly became intractable. In situations with widely spaced points, the IFTS used a single point as the velocity target for several seconds. When the aircraft overflew that point, the IFTS targeted the next point thousands of feet away. These distances were large enough for minor effects such as position error and the curvature of the earth to make the new velocity discontinuous, causing an AIO event. The first quick fix was to increase the allowable trajectory size and increase the number of points in straightaways, and to avoid engaging the IFTS until close to a setup orbit to limit the length of the straightaways.

Typically, the AMM software expects each unique trajectory to be sent to the AMM once. The aircraft autonomously follows the trajectory to its end, at which juncture the automation in control sends a new trajectory as appropriate. In this case minor differences between target and actual velocities compounded over the course of an FPM run, which led to the aircraft being several seconds off target by the end of a test (as depicted in Fig. 5). Additionally, the FPM trajectories were far longer than those typically used by Sikorsky, and often contained more points than the AMM could accept at once. The first solution was to periodically trim the trajectory to the point nearest in the future and re-send the trimmed trajectory to the AMM several times per second. Since the target velocities for these sub-trajectories were re-calculated based on the current aircraft state every time the trim happened, the MW could adjust to velocity under- or overshoots in real time. This method, however, led to the first appearance of lateral AIO. The AMM very smoothly followed a series of points, but there was always a small “bump” when a new trajectory was sent as the software adjusted to its new target. In normal (non-IAS) operation, this condition was acceptable, because new trajectories were not sent often. The MW spaced its trajectory points evenly every three seconds, so the trimming algorithm trimmed off one point approximately every three seconds as time advanced. The AMM therefore had to adjust to a new target as soon as it had settled into tracking its previous target. The new first point was three seconds away from the aircraft, whereas its previous target was directly in front of the aircraft. This problem was exacerbated during turns or climbs, as the discrete points in the trajectory representation necessitated storing turns as essentially a series of consecutive line segments rather than as a continuous curve. The AMM caused the aircraft to jerk laterally when it received a new initial trajectory point non-colinear with the line it was pursuing, and the further off, the larger the AIO.

The solution to this problem was to linearly interpolate between trajectory points during the trim to create a new first point, so that the new first point was approximately three seconds away from the aircraft. The distance to each new trajectory from the aircraft stayed relatively constant, rather than decreasing then suddenly increasing, minimizing the sudden adjustment AMM had to make to address the lateral AIO.

Data Observations: An example of a cyclical lateral AIO is found in Fig. 5 within the 0.2-0.4-Hz right roll angle range of +/- 6 deg. The observed AIO was neutrally stable in amplitude without convergence or divergence. The observed AIOs were independent of climb, descent, accelerations, or decelerations that occurred during a right turn. The pilots experienced a lateral motion from left to right during the AIO event (identified in red) commensurate with the negative aspect of the roll angle. The dihedral effect was observed in the pitch access as a response to the multiple step inputs in the roll axis (see Appendix Fig. A-1).

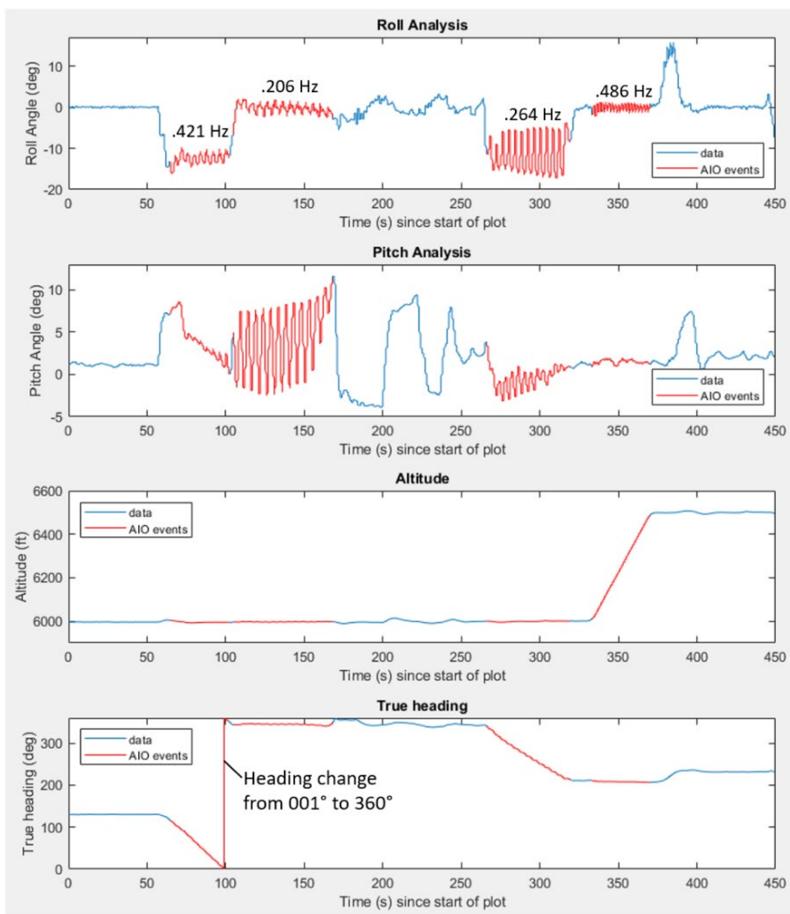

Fig. 5 Example of lateral automation-induced oscillations.

B. Pitch Automation-Induced Oscillations

The second AIO encounter type observed was in the pitch axis in which a traditional “porpoising” was established after rate-limiting commands were sent from the NASA MW to the flight control computer. The oscillations were undetected in the fixed-base simulator because the pilots were unaware at the time.

1. Long Hz 40 - 144 kts Oscillation Time Command

Flight Deck Observations: Researchers observed unstable non-predictable command inputs from the flight control computer. The NASA MW limited aircraft trajectory commands in groundspeed to within a range of 44 - 144 kts authority. Groundspeed was utilized because the NASA MW prioritized the estimated time of arrival as a hard requirement and baselined aircraft performance calculations on groundspeed in lieu of indicated, true, or calibrated airspeed. Due to the system calculations exclusively in groundspeed, the resulting MW high-end limitation command of 144 KGAS was limited by the Sikorsky AMM system limit of 120 KIAS. There was no low-end indicated airspeed command limit. As such, the autonomy commanded 44 kts groundspeed with a tailwind which resulted in indicated speeds at and below 20 kts, well below the turn coordination of the airframe. As a result the aircraft entered effective translational lift vibrations and yawing at altitude when trying to maneuver. This condition was monitored closely, and the test point was terminated when aircraft yawing in descent/deceleration began to increase.

Ground Control Observations: During the flight tests, winds aloft of 20 kts or more were common, and AIO would most likely occur if the commanded encounter setup trajectory began in a headwind. To maintain the trajectory time conformance, the speed of the aircraft had to be modulated, but this MW commanded speed modulation sometimes encountered aircraft speed limits, especially when initiating the command into a headwind (required the groundspeed of the aircraft to increase). This non-linear speed limit hysteresis then led to the observed AIO limit cycle. The larger the headwind, the larger the difference between groundspeed and airspeed, thereby increasing the likelihood of encountering airspeed-related limits while commanding groundspeed [3].

The pitch AIO was caused by instantaneous large changes in the velocity commanded by the IFTS. The AMM struggled to converge after any discontinuities in the velocity; the larger the change, the longer it took to converge. The AMM also took longer to converge during turning or climbing maneuvers. This problem was further exacerbated by non-optimized gains in the PID controller implemented in the IFTS to solve this problem. This controller experienced large integrator windup, which caused it to become under-dampened and resulted in limit cycling. There were three main fixes that allowed satisfactory resolution for the AIO problems: closer spacing of trajectory points, synthesizing the target point for velocity calculation by interpolating between trajectory points, and ad-hoc tuning of the PID controller.

In the IFTS, trajectories are represented as a series of points in time and space; however, the AMM accepts commanded trajectories as a series of points in space such that each point has an associated velocity. The fourth dimension of the AMM is velocity, whereas the fourth dimension of the IFTS is time. Initially, the trajectory was converted from IFTS to AMM format using the time and space deltas between each point, calculating the continuous velocities for the entire trajectory the moment it was loaded. Minor differences between the actual and commanded velocities, however, compounded over the course of (for example) a fifteen-minute FPM run, causing the aircraft to exceed the allowed time error.

This condition required continuous updates to the commanded velocity based on the difference between commanded and actual velocity and adapting to errors in real time. In order to determine what the actual velocity should be at any time, it was necessary to determine where along the trajectory the aircraft should be, based on the current timestamp. This determination was made by comparing the nearest point in the future to the predicted time to reach that point at the current velocity as compared to the desired time taken, and a velocity increase or decrease was then commanded as appropriate.

This initial solution proved to be incredibly sensitive to minor under- or overshoots in velocity. The automation overcorrected in greater and greater amounts, leading to an AIO event unless velocity convergence was perfect. The next, more robust, solution implemented was a PID controller. This controller allowed the commanded velocity to smoothly react to rapid changes in the distance to target. Interpolation between the target point and surrounding points to identify where the aircraft should be in exactly three seconds, was then fed this interpolated distance to the controller. This number of seconds was chosen through experimentation to give the best results. The PID controller was much better at stabilizing on a velocity when the distance it was trying to maintain remained relatively constant, instead of rolling over points repeatedly as in the previous method.

Once a simulator that could emulate AIO was acquired, the software team was able to tune the controller. The debugging prevented some AIO events that were experienced in previous flight events. The controller was experiencing integrator windup due to the same compounding error that prevented calculation of velocity along the entire route at load time. Removing the integrator gain, and tinkering with the proportional and derivative gain, allowed the controller to attenuate AIO to negligible levels. Since headwinds played a role in the AIO limit cycle initiation and magnitude, a correction was made to procedures for the tablet and ground control station (using their displays that presented winds aloft speed and direction) to provide instructions to the aircrew for when to trigger the MW to begin computing or issuing run-in trajectory commands.

Data Observations: An occurrence of a divergent pitch AIO in which the pilots manually disengaged the automation system to maintain the safety of the flight is shown in Fig. 6. The pitch AIO identification threshold for the automation was +/-2 deg (noted by an orange dashed line). The first exceedance was -6 deg, followed by an immediate +10 deg, which nearly repeated again. Next, the pitch oscillated to +9 deg with a subsequent -8 deg and a +8 deg pitch upward. Each frequency sweep occurred within 10 s. Due to the increasingly unstable nature of the oscillations (diverging: -6,-8,-9 deg), the pilot entered the loop by disengaging the automation (green dot) and manually leveled the wings, torque, and targeted 100 kts airspeed.

The Target Velocity (kts) graph in Fig. 6 shows the poorly tuned PID controller resulting in rate limiting velocity commands during the AIO event. Target velocity is the control vector to eliminate time error, and it showed relatively rapid (several times a second) 50-100 kts oscillations while the average target velocity limit cycled over the course of dozens of seconds. The result of the commands are seen in the AMM Time Error graph in Fig. 6. The AMM was unable to respond to the rapid changes in target velocity and continued exceeding maximum time error limits.

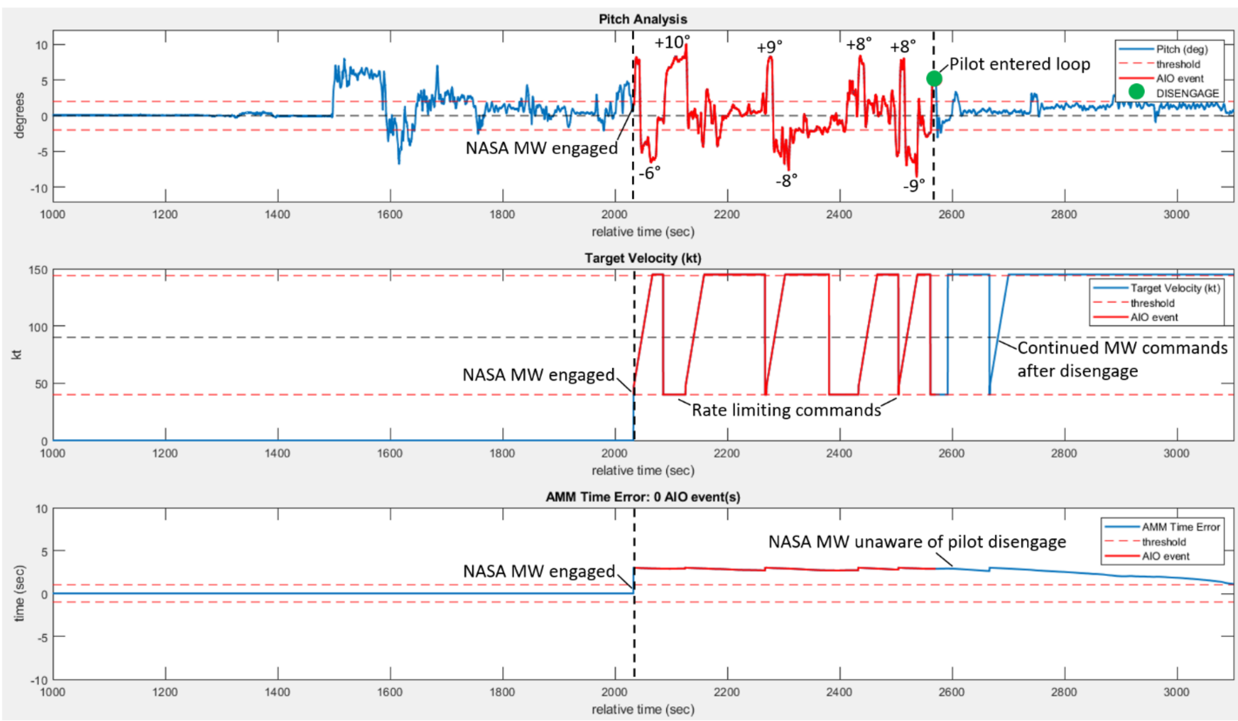

Fig. 6 Example of automation-induced oscillations: Pitch oscillations.

C. Summary of Findings

Low-fidelity simulations of the interactions between the NASA autonomy algorithms and the Sikorsky AMM did not predict AIO in the emergent behavior; however, high-fidelity simulations that incorporated the Sikorsky GenHel flight dynamics model of the Sikorsky helicopter did predict that behavior. This result strongly indicates that the mass properties of the aircraft play a key role in AIO. Furthermore, oscillations were found to be present in the data ahead of AIO manifesting itself in the aircraft control system and being perceived by the pilots. In particular, the time-error parameter, which was a measure of lag in the system, was the first to manifest these oscillations, which are likely related to phase margin in the frequency-domain. These factors indicate that it would be possible to build a platform- agnostic adaptive controller that could automatically detect and tune out non-linear AIO in real-time operations. [6]

V. Conclusion

Emerging technological advancements in AI integration of airspace management and dynamic aircraft routing will continue to increase. However, the same undesired aircraft oscillation anomalies will manifest in the future flight environment. Lessons learned from the National Aeronautics and Space Administration Integration of Automated Systems flight tests have resulted in the development of a flight control tool that is tunable related to flight envelope, complex interactions, and time offset. This tool could predict automation-induced oscillations (AIO) to control through autonomy algorithms. The tool, however, should be calibrated specifically for each aircraft as an automated, dynamic function. Given the nascent state of AIO, there is a need to develop and test an AIO controller for the future.

A. Research Questions for Future AIO Evaluation

1) *What does AIO look like?*

   An uncommanded climb/descent, turn, speed change, coupling or navigation mechanism

2) *What are the root causes for AIO and how to predict it?*
   a. The AIO observed in this testing was predictable only when using high-fidelity simulations that included the inertial dynamics and attenuation of the aircraft
   b. Attenuation of AIO needed to be tuned specifically for a given aircraft
   c. The AIO may be an interaction between predicted aircraft state and actual aircraft state due to inertially-driven latencies in the response of the aircraft, however, this relationship has not been formally established

3) *How does one evaluate AIO?*
   a. Cyclical (i.e., coding) and non-cyclical (i.e., environmental response/condition)
   b. Phase / gain margins

4) *What is acceptable AIO?*

   Determine acceptability for developmental testing, operational testing, vehicle limitations, and passenger comfort

5) *What role does a test pilot have when evaluating AI/ML airspace/aircraft integration?*
   a. What is commanding a system?
   b. Where is it located?
   c. Is 4DT time-based or velocity-based for waypoint following?
   d. What authority does each system have?
   e. What aircraft mode changes are required? (cruise, low speed transition, hover, taxi)
   f. What are the system mode limitations? (airspeed: cruise, transition, hover, et cetera)
   g. What are the learning rates for AI systems? (learning while in-flight versus with test pilot)
   h. When to abort or continue autonomous flight? (knock it off or ride it

Appendix

Table A-1 Automation-induced oscillations: Summary.

Aircraft	No. of AIO events	Mean duration (min)	Min. Wind Speed (kn)	Max Wind Speed (kn)	Max Wind Offset (deg) *relative to aircraft heading*
OPV	1	6.00	8	8	31
SARA	2	2.50	4	9	17
SIM	1	7.00	0	0	0
OPV	2	2.50	1	3	188
SARA	4	1.75	1	2	279
SARA	2	2.50	4	6	237
OPV	0	0.00	0	0	0
OPV	1	7.00	15	15	198
SARA	2	4.00	9	13	203
SARA	1	3.00	13	13	203
SARA	1	2.00	13	13	189
OPV	2	8.50	17	19	357
SARA	1	5.00	23	23	6
OPV	2	7.00	14	22	263
OPV	2	5.00	17	19	261
OPV	1	2.00	17	17	253
OPV	1	2.00	9	9	224
SARA	5	3.80	20	26	264
SARA	1	3.00	18	18	264
OPV	1	2.00	19	19	300
OPV	1	4.00	18	18	300
OPV	1	2.00	6	6	290
SARA	2	3.50	15	15	290
SIM	4	1.50	0	9	353

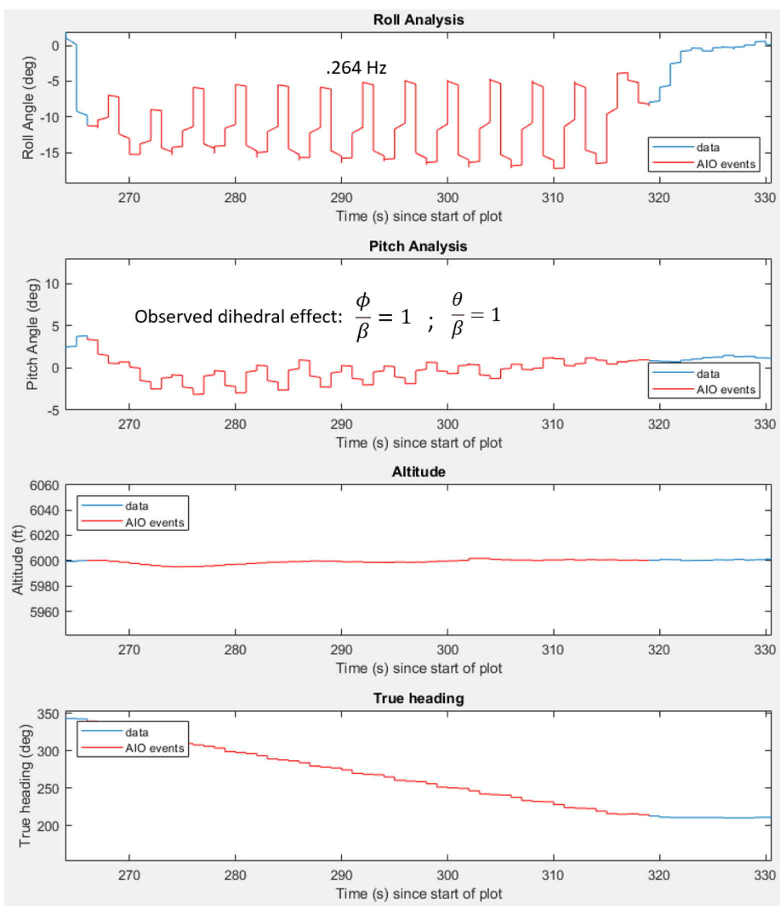

Fig. A-1 Automation-induced oscillation lateral ratcheting example.

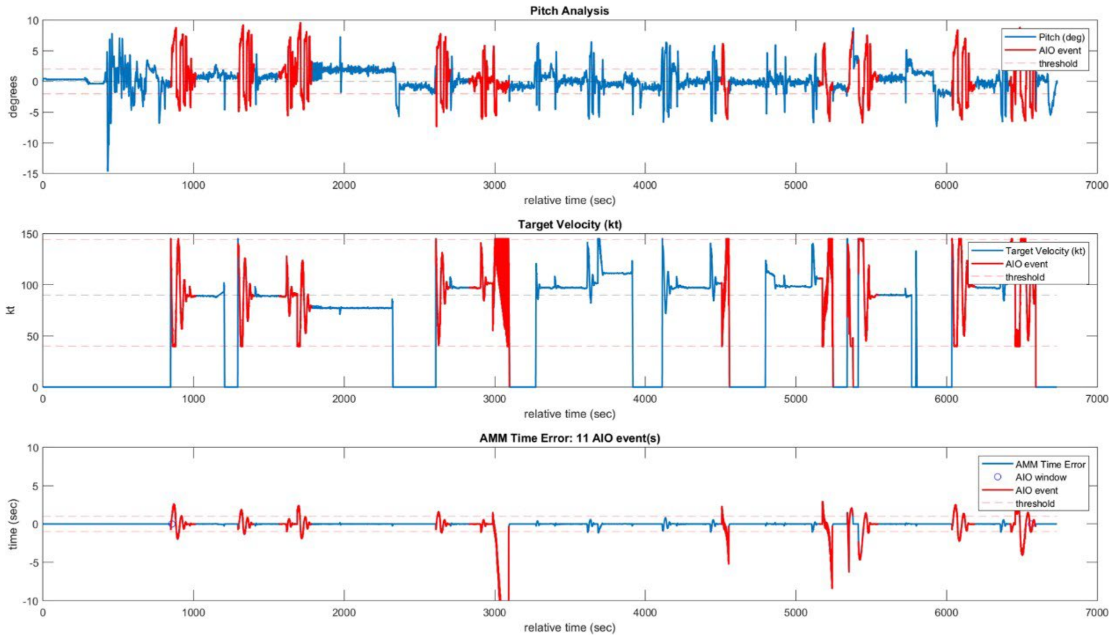

Fig. A-2 Automation-induced oscillations event of October 23, 2023.

Acknowledgments

The authors thank Ethan Williams, William 'Bill' Fell, Jan Scofield, Daniel Eng, Scott 'Jelly' Howe, Nancy Baccheschi, Starr Ginn, Gerrit Everson, and the entire Lockheed Martin Sikorsky team for their support, dedication and contributions to the research.

References

[1] McRuer, D. T., *Pilot-Induced Oscillations and Human Dynamic Behavior,* NASA Contractor Report 4683, 1995.

[2] Sharma, V., and Wing, D. J., *Dynamic Path Planning Automation Concept for Advanced Air Mobility,* NASA TM-20220009974, 2022.

[3] Baccheschi, N., Scofield, J., Williams, E., Guion, A., Monk, K., Fettrow, T., et. al., "Integration of Automated Systems Test Campaign," Airworthiness Assessment Memo number AAM-NC-129-001 (to be published).

[4] Department of Defense Interface Standard, Pilot-Induced Oscillations: Their Cause and Analysis, Washington, D.C.: https://apps.dtic.mil/sti/tr/pdf/AD0481994.pdf

[5] National Academy of Sciences, "Aircraft-Pilot Coupling Problems: Definitions, Descriptions, and History," in *Aviation Safety and Pilot Control: Understanding and Preventing Unfavorable Pilot-Vehicle Interactions,* National Academy Press, Washington, D.C., 1997, pp. 161, 162.
doi: 10.17226/5469.

[6] Clements, K., "Time Domain Stability Margin Assessment," 2016 AIAA Young Professionals Symposium, October 2016. URL: https://ntrs.nasa.gov/api/citations/20160013364/downloads/20160013364.pdf