AIAA SciTech Forum
7-11 January 2019, San Diego, California
AIAA Scitech 2019 Forum

10.2514/6.2019-0984

An Analysis of Air Traffic Management Concepts of Operation Using Simulation and Formal Verification

Lanssie Mingyue Ma [*]
Georgia Institute of Technology, Atlanta, GA, 30313.

Adam Houser [†]
University at Buffalo, Buffalo, NY 14260.

Karen M. Feigh [‡]
Georgia Institute of Technology, Atlanta, GA, 30313

Matthew L. Bolton [§]
University at Buffalo, Buffalo, NY 14260

Emerging air traffic management operations attempt to improve the capacity of the air transportation system. However, system complexity can make it difficult to predict system performance and find potentially unexpected problems. Agent-based simulations have been used to evaluate different function allocations for air traffic management procedures in terms of their implications for flight performance and human agent taskload. However, simulations can miss unexpected situations. We have developed a method that uses formal verification (a means of proving whether or not a model satisfies specific properties) with agent-based simulation to completely explore the space around simulated air traffic scenarios. Interesting conditions found with verification can be more deeply analyzed with the simulation. Here, we use this method to account for limits of human operator taskload, action timing, and action prioritization to find problems with previously evaluated air traffic management scenarios. We found that there are function allocations where the considered human limits can significantly impact performance. We discuss these results and directions for future research.

Nomenclature

ATC	the amount of actions that an agent is performing.
Authority	delineate which actions an agent is asked to perform.
Autonomy	delineate which actions an agent can perform independently.
Responsibility	delineate which outcomes an agent will be accountable for in an organizational, regulatory or legal sense.
Concepts of Operation	define the authority and responsibility of agents to complete actions while function allocation defines the division of labor between humans and automated systems.

I. Introduction

With the introduction of the Next Generation Air Transportation System (NextGen) and the Single European Sky Air Traffic Management, major changes are coming to the way that air traffic management is conducted. These air traffic management concepts are attempting to use different function allocations between air traffic controllers, pilots, and automated agents to better distribute taskload between the different agents to hopefully increase the capacity of the air transportation system while improving system safety and reliability [1–4].

Because the new air traffic management concepts rely on operations being executed more efficiently to increase air transportation capacity, the timing of actions is critical. A human's ability to execute his or her actions efficiently will

[*]Ph.D. Candidate, Daniel Guggenheim School of Aerospace Engineering, 270 Ferst Dr.,
[†]Ph.D. Candidate, Industrial and Systems Engineering, 407 Bell Hall,
[‡]Associate Professor, Daniel Guggenheim School of Aerospace Engineering, 270 Ferst Dr., [AIAA MEMBER GRADE HERE]
[§]Assistant Professor, Industrial and Systems Engineering, 407 Bell Hall

largely depend on his or her taskload (a measure of the number of tasks a human operator is expected to perform at a given time) and how he or she prioritizes those tasks [5–7].

It is infeasible to use real-world experiments to evaluate the implication of different function allocations on system performance within a system as complex as ATM, especially early-on in the design process. However, evaluations are critical because it can be difficult to predict all the possible conditions that could produce excessive taskload for human agents and how these operating conditions could impact air traffic performance. Human subject experiments, like those found in [8], involve part-task simulation to evaluate subjective workload in different situations. However, these only allow the impact of taskload to be evaluated for a narrow set of conditions and, even at this fidelity, level these simulations requires significant resources to run. Model-based techniques, such as agent-based simulation, have been employed to evaluate how different function allocations impact performance and taskload in a larger variety of contexts [9–11]. While more flexible than using human subject and real-world experiments, simulation is also limited in that it may take thousands of simulation runs to characterize an operational concept. Even with such efforts, the complexity of the system can mean that simulations can miss critical conditions that could result in unexpected outcomes.

An alternative approach can be found in the field of formal methods [12, 13]. Formal methods are tools and techniques for mathematically proving properties about models of systems, a process known as formal verification. Formal verification has advantages over simulation in that it provides a complete analyses of the system model and can thus find system interactions and problems not found by other approaches. However, this completeness comes at the cost of scalability. Specifically, the size of the system model grows exponentially as concurrent elements are added to it. As a result, abstraction techniques must be used to keep model size under control [14]. In spite of this, these abstractions can result in important details being removed from the model that could be important for identifying problems. As a result, formal methods are also limited.

In previous work, we introduced a method that combines the high-fidelity analysis capabilities offered by agent-based simulations (based on Work Models that Compute (WMC) – discussed subsequently) and the completeness of formal verification [15, 16]. In this, analysts can run a high-fidelity simulation scenario and then use formal verification to iteratively explore the space around to find interesting conditions that might have been missed otherwise. Conditions found with this method can then be used to create modifications of the original scenario that can be re-explored with the simulation.

In the work discussed here, we applied this method to the analysis of air traffic management concepts of operation to evaluate the way that different function allocations can impact human taskload and system performance.

II. Background

A. Work Models that Compute

Work Models that Compute (WMC) is a simulation framework that dynamically models complex, multi-agent concepts of operations and work domains [17]. WMC models the collective work of multiple agents [18]. It has two parts: a work model that describes the domain's work and an engine that simulates the work model [17]. A work model has three elements: agents, actions, and resources. Resources represent the elements of the work environment that the agent can interact with. Actions are linked to specific agents at runtime and represent atomic behaviors that change the values of resources. The work model can also describe each action's frequency, the resources it needs or manipulates, and the agents it is linked to [19] (though not all of these concepts are represented in all scenarios). Agents are used to organize actions. Agent taskload can be tracked and measured based on the number of actions assigned to them at any given time [18].

A scenario combines work models, agents, actions, and resources into a simulation. A scenario is then run to generate an action trace (detailed timing descriptions of the actions executed by each agent over the course of the simulation) and other domain relevant outputs such as function allocation metrics [11]. The simulation engine uses a hybrid timing mechanism that allows for elements of continuous time and event-based simulation. This enables WMC to simulate both dynamic systems (such as aircraft dynamics) and event-based agents (such as pilot models) [19, 20].

In previous work by Feigh and Pritchett et al., WMC has been used to develop simulations to evaluate function allocation issues in the air transportation domain [3, 11, 21]. Specifically, scenarios have been created for evaluating the Traffic Collision Avoidance System (TCAS); studies using these simulations have assessed how changes in pilot authorities and responsibilities with clearance for visual approaches impact system performance [11]. Most important to the presented work is research that used WMC simulation to explore different function allocations for NextGen air traffic management concepts with different levels of authority and responsibility between agents [3, 21]. In these

Table 1 The FA1FA1 and FA3FA3 Function Allocations [3, 21]

Function Groups	FA1FA1	FA3FA3
Vertical profile control	ATC	Pilots
Aircraft configuration management	ATC	Pilots
Lateral control	ATC	Pilots
Speed control	ATC	Pilots
Lateral profile management	ATC	ATC
Vertical profile management	ATC	ATC
Speed management	ATC	ATC
Non-nominal situation management	ATC	ATC

analyses, authority indicated which agent was required to execute a task and responsibility indicated which action was accountable for the outcome of a task. The scenarios in these analyses included three aircraft agents and an air traffic controller agent interacting in a merging descent landing task.

For example, two of the conditions considered in these analyses [3, 21] were FA1FA1 and FA3FA3. These scenarios feature three aircraft and one ATC agent working to land on the same runway. The function allocations represent the different division of labor to these agents and evaluating the effect on the nominal landing sequence. In these scenarios, the numbers in the name indicate the allocations of authority and autonomy as shown in Table 1. FA1FA1 represents a scenario where the ground has all of the authority (to carry out the actions) and responsibility (to ensure they are completed). The ATC is primarily responsible for all the task executions to land the aircraft. FA3FA3 represents a more-balanced distribution, where air and ground share equal amounts of responsibility and authority. Unlike the previous scenario, the agent responsible for a task or action also have the authority to execute it. Different function groups, or collections of actions, are given to specific pilots to execute. These differ by the function allocations shown in Table 1.

The analysis of the considered scenarios (including the ones in Table 1) provided key insights into the aggregate taskwork (the total number of actions/tasks performed) and information transfer requirements for different function allocations. For all scenarios, including those in Table 1, all three of the analyzed aircraft were successfully directed to the ground without incident. While useful, only one scenario was run for each function allocation. Further, to evaluate the concepts of operation without confounding factors, agents in these analyses would execute actions as soon as they were assigned. Thus action execution time and prioritization did not impact the results. Accordingly, these simulations did not account for human operator taskload restrictions (how many actions an agent can execute at a time) nor did it account for the variance in action timing this and action priority could produce. Therefore, it is likely that the analysis of these scenarios could have missed critical or interesting conditions that could occur with minor variations in the simulations task order or timing. Because of the complexity of the considered scenario, it would likely be difficult or impossible for an analyst to manually explore scenario variations to find these conditions. Of particular import for the FA1FA1 and FA3FA3 scenarios is the fact that, because action duration and prioritization was not considered, the timing of actions could profoundly impact the results. Formal verification, and its use with simulation, constitutes a different model-based approach that had the potential to address these issues.

B. Formal Verification and its Synergistic Use with Simulation

Formal verification comes from the area of formal methods [12]. Formal methods are tools and techniques for modeling, specifying, and verifying systems. Modeling uses robust mathematical languages to describe the behavior of a target system. Specification mathematically describes the behavior the system is expected to exhibit. Formal verification is the act of mathematically proving whether or not the formal model satisfies the specification.

While there are different approaches to formal verification, one of the most well-known is model checking. Model checking allows formal verification to be performed automatically [13]. In this, a formal system model is described as a collection of concurrently executing state machines. Specifications are usually asserted in a temporal logic. Model checking performs formal verification by using efficient algorithms to exhaustively search through the system model's entire statespace to find specification violations. If one is not found, the model checker returns a confirmation indicating that the model satisfies the specification. Otherwise, the model checker produces a counterexample: a trace through the model that illustrates exactly how the specification was violated.

Formal verification, and particularly model checking, have been used successfully to find and correct problems in a number of computer hardware and software systems. Further, a growing body of literature has shown how these technologies can be used in the engineering of human-computer and human-automation interaction [22, 23]. However, the biggest limitation to the use of formal verification is scalability due to the state explosion problem [13]. Specifically, as concurrent elements are added to the system model, the size of the model grows exponentially. This can lead to a situation where the model takes too long or is too big to verify. As a result, analysts must often use abstraction when constructing a system model to enable tractable analyses [14].

Researchers have found some success in using formal verification synergistically with simulation in order to exploit the exhaustive capabilities of model checking with simulation's ability to represent systems with higher fidelity, while avoiding the limitations of each. Specifically, formal verification is typically used to selectively to evaluate bounded elements of a simulated system [24–27]. Of particular relevance to the presented work is research that has used simulation traces to create formal models that are small enough to avoid scalability [28–30]. The limitation of these approaches is that they only check properties about the actual traces and thus do not account for any system behavior beyond what is already contained within them. In a method we have developed [15, 16], we can use model checking analyses to explore a more complete space around a simulation trace and thus find potentially interesting, i.e. dangerous, operating conditions that were missed in this simulation.

C. Our Approach to Dual Analysis

Our method to the dual use of formal verification and simulation is shown in Fig. 1. This methods gives analysts the ability to use model checking's exhaustive search capabilities to explore the region around simulated air traffic scenarios from WMC. This allows for the discovery of previously unexpected human taskload conditions and/or resource conflicts. The method works as follows [15, 16]:

1) A WMC work model and a scenario (both described above) are run through a WMC simulation. The simulation produces a trace that shows what actions were executed when and by whom.
2) The work model, scenario, and simulation trace are then translated into a formal model representing the simulation over an analyst-specified period of time (a time window). Because the timing of actions and their duration is critical to the manifestation of taskload, the formal model explicitly represents these concepts. This representation can include analyst-defined variance to allow the model checker to explore the performance space around the modeled scenario. The formal model represents each human agent as having a limited capacity of actions that it can execute at any given time (*activeCapacity*) as well as a limited capacity of actions that it is expected to perform in the future (*inactiveCapacity*) that can be specified by the WMC work model or (if it is not in the work model) the analyst. The formal model dynamically determines which actions assigned to agents are based on their priority (higher is better) and execution time (lower is better). In cases where actions have the same priority, the model accounts for situations where each of the possible options is given priority.
3) The translator also generates a set of specification properties designed to find interesting taskload conditions in the model (discussed subsequently).
4) A model checker is used to search through the formal model to find counterexamples (traces) that show how specifications were violated.
5) The traces are translated back into WMC scenarios so that the modified scenario can then be analyzed with followup simulations.

It is important to note that this process can be applied iteratively. Specifically, scenarios can be modified by progressively moving the time window between method applications to the end of the scenarios. Applying the method in this way allows for changes made early in the scenario to be fully considered across the scenario.

A number of different generated specifications can be used with the model checking analyses to find interesting taskload conditions and resource conflicts (see [16]). For the work discussed in this paper, we are only concerned with the one designed to find conditions that avoid overload in taskload:

$$FindNoOverload \models G\neg\left(\left(\begin{array}{c} status = doing \vee status \neq doing \\ \wedge \left(\begin{array}{c} (cardinality(inactive)) \\ \leq agent[i].inactiveCapacity \end{array}\right)\end{array}\right) U\,(globalTime \geq Never)\right). \quad (1)$$

This is a linear temporal logic specification [31] that asserts that the number of inactive actions an agent is assigned should never exceed its capacity. Specifically, this can be interpreted as: for all **G** paths through the model, it should never

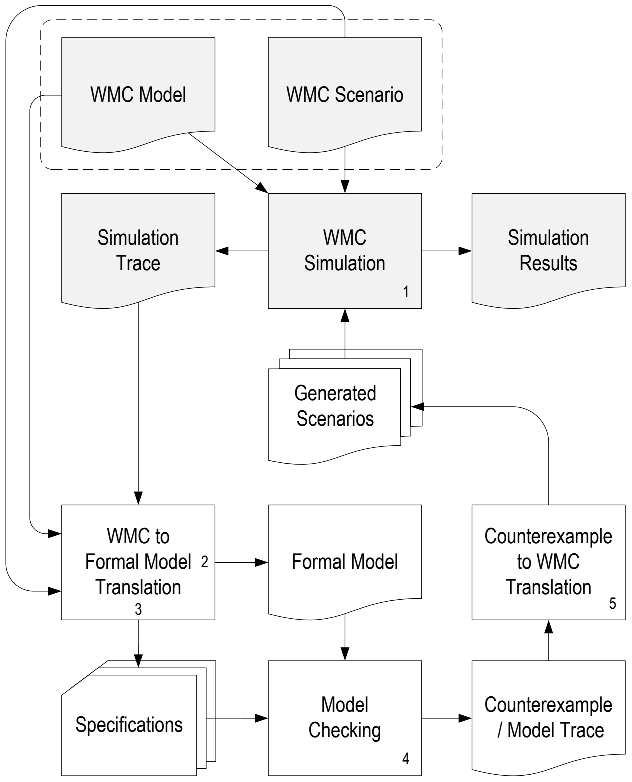

Fig. 1 Flow diagram illustrating the our method for the synergistic use of WMC simulation and model checking [15, 16].

be that case (¬) that the number of inactive actions assigned to the agent (*cardinality*(*inactive*)) should never exceed its capacity (*agent*[*i*].*inactiveCapacity*) until (**U**) past the end of the considered time window (*globalTime* ≥ *Never*). Thus, when this specification is checked, if there is a way to make it to the end of the considered time window without the human being overloaded, the model checker will return a counterexample showing how this occurred.

We implemented this method as a software tool that automates that majority of the steps in Fig. 1. It uses WMC for simulations and the infinite bounded model checker (from the symbolic analysis laboratory (SAL)[32]) for formal verifications.

III. Objective

A manual inspection of the FA1FA1 and FA3FA3 WMC scenarios that were evaluated in [3, 21] reveals that many actions are expected to occur at the same time, a situation that can result in human operator overload. Both scenarios focused initially on how varying function allocation can impact human operator taskload and flight performance in emerging ATC concepts of operation. This coupled with the fact that the simulation did not account for the duration of actions or the prioritization of their execution implies that the performance predictions associated with these analyses are potentially predicting an efficiency of performance that is unrealistic. In the work presented here, we attempted to address this issue by using our dual analysis method to find variations of the scenarios that account for action timing, limits on human operator taskload, and action prioritization to find scenarios that avoid excessive taskload. We then examined the scenarios that resulted from these analyses to understand how function allocation impacts performance of the air traffic management system.

We hypothesized that by delaying when agents execute actions to prevent human operator overload, we would see aircraft spaced too close together. We did not expect to produce problems for FA3FA3 because of the balanced function allocation. However, we did hypothesize that because FA1FA1 was biased in favor of ATC having responsibility and authority for all of the considered actions, the effect of the action delays would be more significant and potentially prevent the landing of at least one of the aircraft.

IV. Methods

A. The Considered Scenarios

The scenarios considered in this analysis were adapted from IJitsma et al.'s [3, 21] work that was developed to analyze different concepts of operation in ATC. These varied the function allocation to the agents involved based on their autonomy, authority, and responsibility. All of the scenarios included three aircraft agents and an ATC agent coordinating in a merging descent landing task. The taskload was effectively allocating different actions between the ATC and aircrafts. These scenarios evaluated various function allocations on nominal and off nominal plane descent patterns. Our scenario only examines cases with nominal execution, which we can view from a side profile (Fig. 4) and a bird's eye view (Fig. 3). Our experiment focused the two scenarios from the original analyses discussed previously, FA1FA1 and FA3FA3 Table 1.

These scenarios were chosen because they did not result in any inconsistent allocation between authority and autonomy [21]. Furthermore, they exhibit a contrast between a balanced function allocation for ATC and the aircraft FA3FA3, and an imbalanced one where ATC is responsible for everything FA1FA1. They have also been the subject of multiple analyses in the WMC literature [3, 21].

In both scenarios, three aircraft are attempting to land at Amsterdam's Schiphol Airport in the Netherlands, see Fig. 2. The first aircraft serves as the leader. It enters from the west and performs an optimal profile descent. The following aircraft enter from the East and perform in-trail and merging interval management, where they ultimately follow the first aircraft into an optimum profile descent. More details can be found in [21].

In the previous analyses of these scenarios [3, 21], the agents were treated as being "perfect" in that they execute actions as soon as they were assigned. Thus, action time and priority were not a factor. This allowed the concepts of operation to be isolated and analyzed. However, this failed to account for the impact that human taskload could have on performance. We considered these factors in our application of the method.

We consulted with aviation subject matter experts to identify the nominal times it would take agents to execute actions. We further prioritized these actions by classifying the intent of each action within an *aviate*, *navigate*, and *communicate* framework from the naval aviators guide for pilots [21]: *aviate*, *navigate*, and *communicate*. Table 2 summarizes these parameters for the actions included in the analyses. Nominal flight trajectories and latitudes can be viewed in Figs. 3 and 4.

Table 2 Action parameters

Action Name	Priority	Execution Times
direct_to_WP	high	5
set_flaps_speedbrakes	high	7
clear_for_descent	medium	7
manage_WP_progress	medium	7

Note that each of these actions can exist for all four of the agents included in the analysis.

B. Apparatus

All analyses reported in this paper were conducted on a computer workstation configured with a 3.7GHz Intel Xeon® quad-core processor and 128GB of RAM. This ran our software implementation of the method along with the associated WMC simulation and SAL's infinite bounded model checker.

C. Method Application

We applied our method (Fig. 1) to the two considered scenarios as follows:

1) A WMC scenario was simulated to completion using the baseline configuration of action priorities and execution time parameters (no action durations and no priorities). This generated an action trace file that captured each of the combined ~150,000 actions executed by all three aircraft in the unmodified simulation.
2) The WMC action trace and XML files generated from the simulation run were then imported into the software implementation of the method.

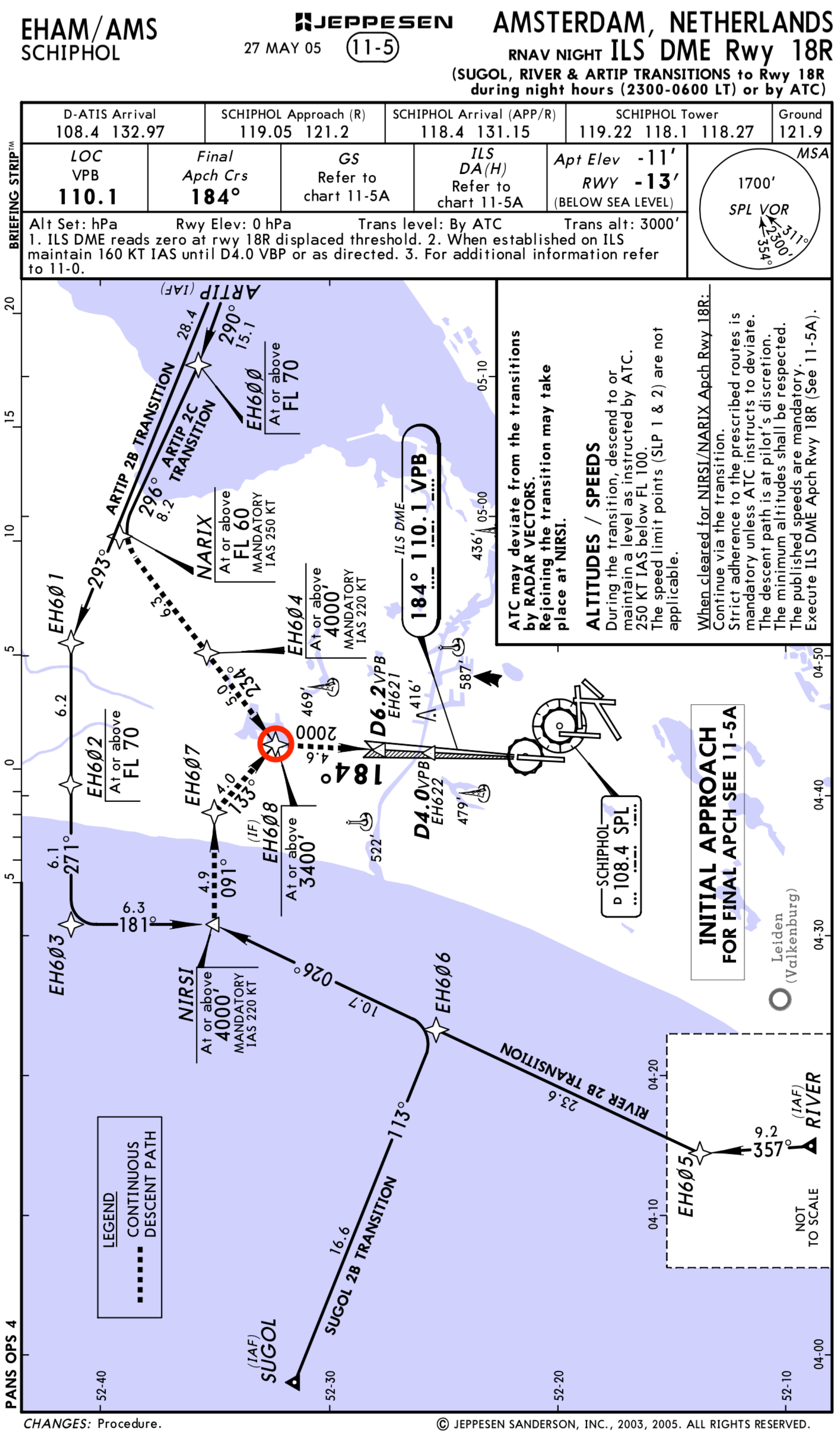
EHAM/AMS
SCHIPHOL
JEPPESEN
27 MAY 05
11-5
AMSTERDAM, NETHERLANDS
RNAV NIGHT ILS DME Rwy 18R
(SUGOL, RIVER & ARTIP TRANSITIONS to Rwy 18R during night hours (2300-0600 LT) or by ATC)
D-ATIS Arrival 108.4 132.97
SCHIPHOL Approach (R) 119.05 121.2
SCHIPHOL Arrival (APP/R) 118.4 131.15
SCHIPHOL Tower 119.22 118.1 118.27
Ground 121.9
BRIEFING STRIP™
LOC VPB 110.1
Final Apch Crs 184°
GS Refer to chart 11-5A
ILS DA(H) Refer to chart 11-5A
Apt Elev -11′
RWY -13′
(BELOW SEA LEVEL)
MSA
1700′
SPL VOR
311°
2300′
354°
Alt Set: hPa Rwy Elev: 0 hPa Trans level: By ATC Trans alt: 3000′
1. ILS DME reads zero at rwy 18R displaced threshold. 2. When established on ILS maintain 160 KT IAS until D4.0 VBP or as directed. 3. For additional information refer to 11-0.
ARTIP (IAF)
28.4
290°
15.1
ARTIP 2B TRANSITION
ARTIP 2C TRANSITION
296°
8.2
EH6Ø0 At or above FL 70
NARIX At or above FL 60 MANDATORY IAS 250 KT
293°
EH6Ø1
9.3
EH6Ø4 At or above 4000′ MANDATORY IAS 220 KT
6.2
EH6Ø2 At or above FL 70
5.0
224°
2000
4.6
184°
D6.2 VPB EH621
D4.0 VPB EH622
ILS DME 184° 110.1 VPB
EH6Ø7
4.0
133°
(IF) EH6Ø8 At or above 3400′
4.9
091°
6.1
271°
6.3
181°
EH6Ø3
NIRSI At or above 4000′ MANDATORY IAS 220 KT
026°
10.7
EH6Ø6
113°
16.6
SUGOL 2B TRANSITION
(IAF) SUGOL
RIVER 2B TRANSITION
23.6
EH6Ø5
9.2
357°
(IAF) RIVER
469′
416′
587′
436′
479′
522′
SCHIPHOL D 108.4 SPL
Leiden (Valkenburg)
INITIAL APPROACH FOR FINAL APCH SEE 11-5A
NOT TO SCALE
LEGEND CONTINUOUS DESCENT PATH
ATC may deviate from the transitions by RADAR VECTORS. Rejoining the transition may take place at NIRSI.
ALTITUDES / SPEEDS
During the transition, descend to or maintain a level as instructed by ATC. 250 KT IAS below FL 100. The speed limit points (SLP 1 & 2) are not applicable.
When cleared for NIRSI/NARIX Apch Rwy 18R:
Continue via the transition. Strict adherence to the prescribed routes is mandatory unless ATC instructs to deviate. The descent path is at pilot's discretion. The minimum altitudes shall be respected. The published speeds are mandatory. Execute ILS DME Apch Rwy 18R (See 11-5A).
52-40
52-30
52-20
52-10
04-00
04-10
04-20
04-30
04-40
04-50
05-00
05-10
PANS OPS 4
CHANGES: Procedure.
© JEPPESEN SANDERSON, INC., 2003, 2005. ALL RIGHTS RESERVED.

Fig. 2 Aircraft flight path to Amsterdam's Schiphol Airport [21]

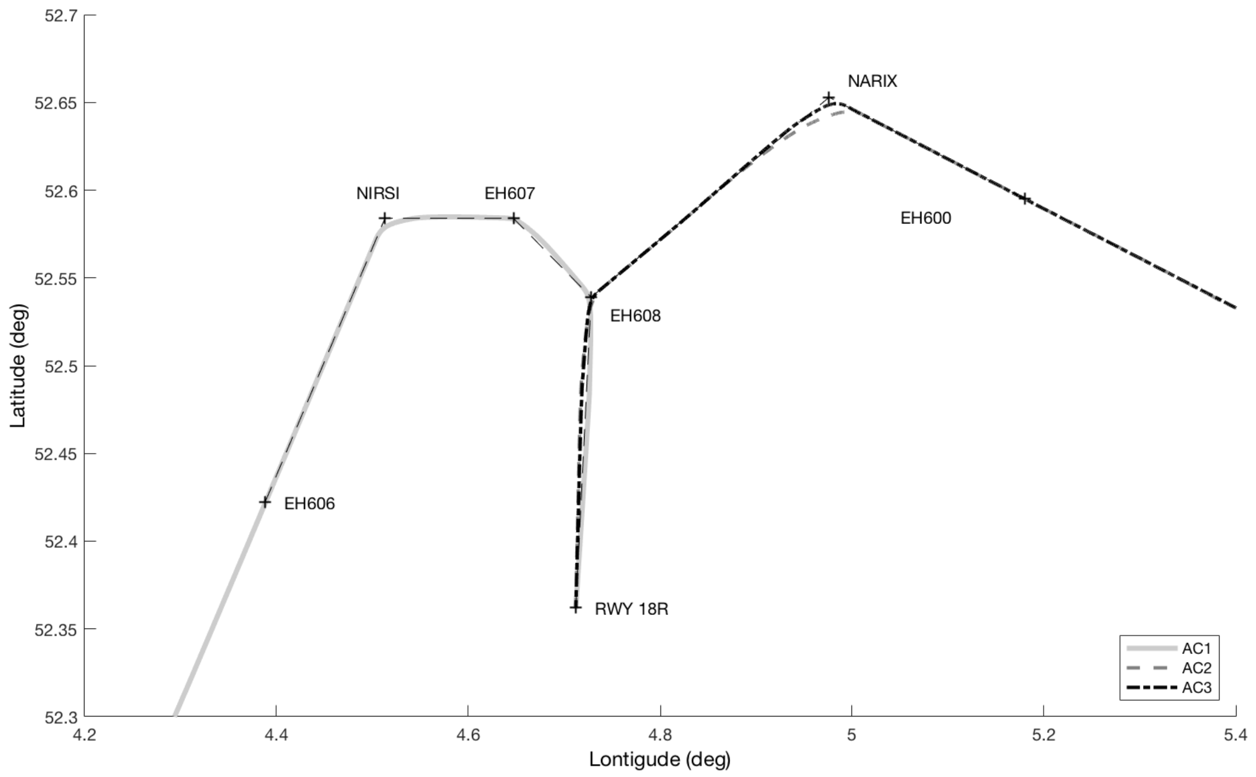

Fig. 3 Nominal top view of plane descent scenario

3) The software implementation read in the contents of the action trace file, parsing through the raw action trace output and translating each entry (containing an action execution time, its name, the agent to which it belonged, and its programmed next update time) into a table. Using this format, we then manually identified a time window to more fully explore with formal verification. Because actions tended to collocate throughout the simulation trace, we were able to select relatively small temporal windows.
4) We used the software tool to isolate the agents and actions that participated in the analyzed section of the trace as dictated by the time window.
5) Priorities and time durations were automatically assigned to each included action based on the values identified in Table 2.
6) The software program used these parameters to automatically create a formal model in the input language of SAL. This file represented the behavior of the agents from the WMC simulation with the added variance and nondeterminism associated with the newly included action timing and priority values. Note that in the presented analyses, all human agents had the ability to execute two actions at a time and a capacity for remembering five actions that had been assigned to it, but not yet completed. This resulted in humans having the capacity to remember 7 actions at a given time [33]. This file also contained generated specifications used for finding interesting taskload conditions in the formal model. This included specifications using the pattern from (1) created for each agent considered in the analyses.
7) SAL's infinite bounded model checker was run to check the specification properties (1) for each agent. The resulting counterexamples identified how overload could be avoided by spacing out each agent's actions.
8) These counterexamples were then automatically processed by our software tool. This created a new version of the WMC scenario with the action timings found in the counterexample.
9) The new scenario was run through the WMC simulation to produce a new action trace and simulation outputs. These were examined and the process was repeated (starting at step 2) as necessary to iteratively move the time window through the scenario.

The actual process was a cycle of steps over fourteen to sixteen distinct iterations on each original scenario. Actions tended to group together, resulting in rescheduling groups. One hundred seconds was ultimately used on the upper limit

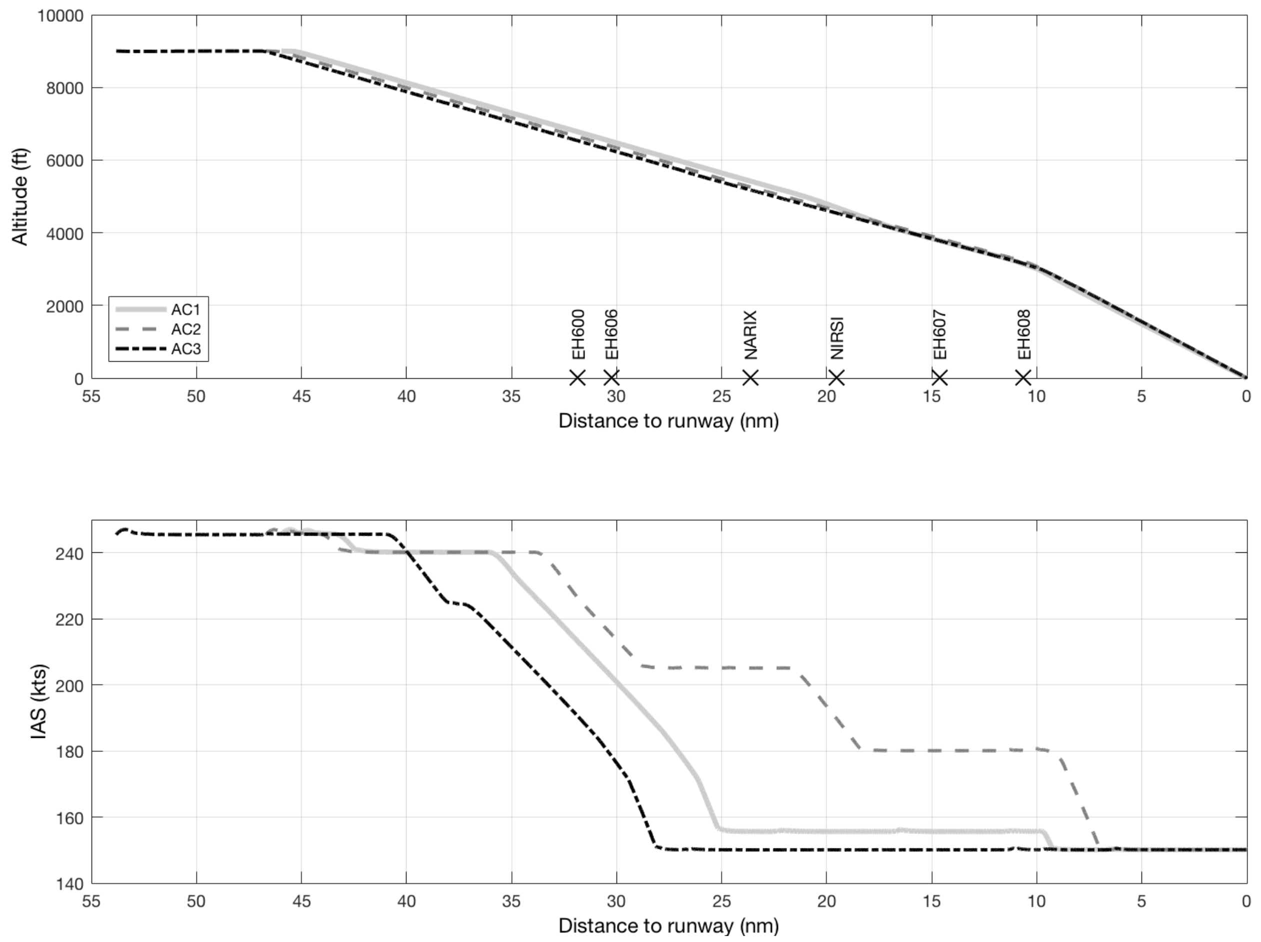

Fig. 4 Nominal side view of plane descent scenario

(though lower limits were possible if compelling results were produced before this) of the scenario because this was where the two following aircraft were nominally given their clearance instruction by checking ATC. An aircraft that is not ready to receive its descent clearance may need to perform a go-around. Thus, at the 100 second position we extracted the GPS coordinates of the aircraft and measured the distances between them. We then used this measure to determine how far off track each aircraft was (as compared to its position in the original, unmodified scenario).

V. Results

Below we present results for the two analyzed scenarios of aircraft descent.

A. FA1FA1 Scenario

The first 58 seconds of the FA1FA1 scenario were ultimately processed with our method over 8 iterative steps. This was done because significant problems arose in the simulation long before reaching a 100 seconds threshold. Here, we were able to show results of a worst case scenario after nearly half the time for the full 100s threshold. Specifically, as actions were delayed to prevent taskload of the air traffic controller, the aircraft spread out more than in the original scenario. This ultimately resulted in a situation where (considering the full scope of the method-produced simulation scenario) the first and third aircraft were successfully able to land. However, the second aircraft would have caused a loss of separation and was therefore forced to perform a back off and reinsertion into the approach maneuver in the modified scenario.

The visualization of the flight path is shown in Fig. 5. This illustrates the magnitude of the delay experienced by the third aircraft in the modified scenario. In this, green markers indicate the original, unmodified location of aircraft 3 at

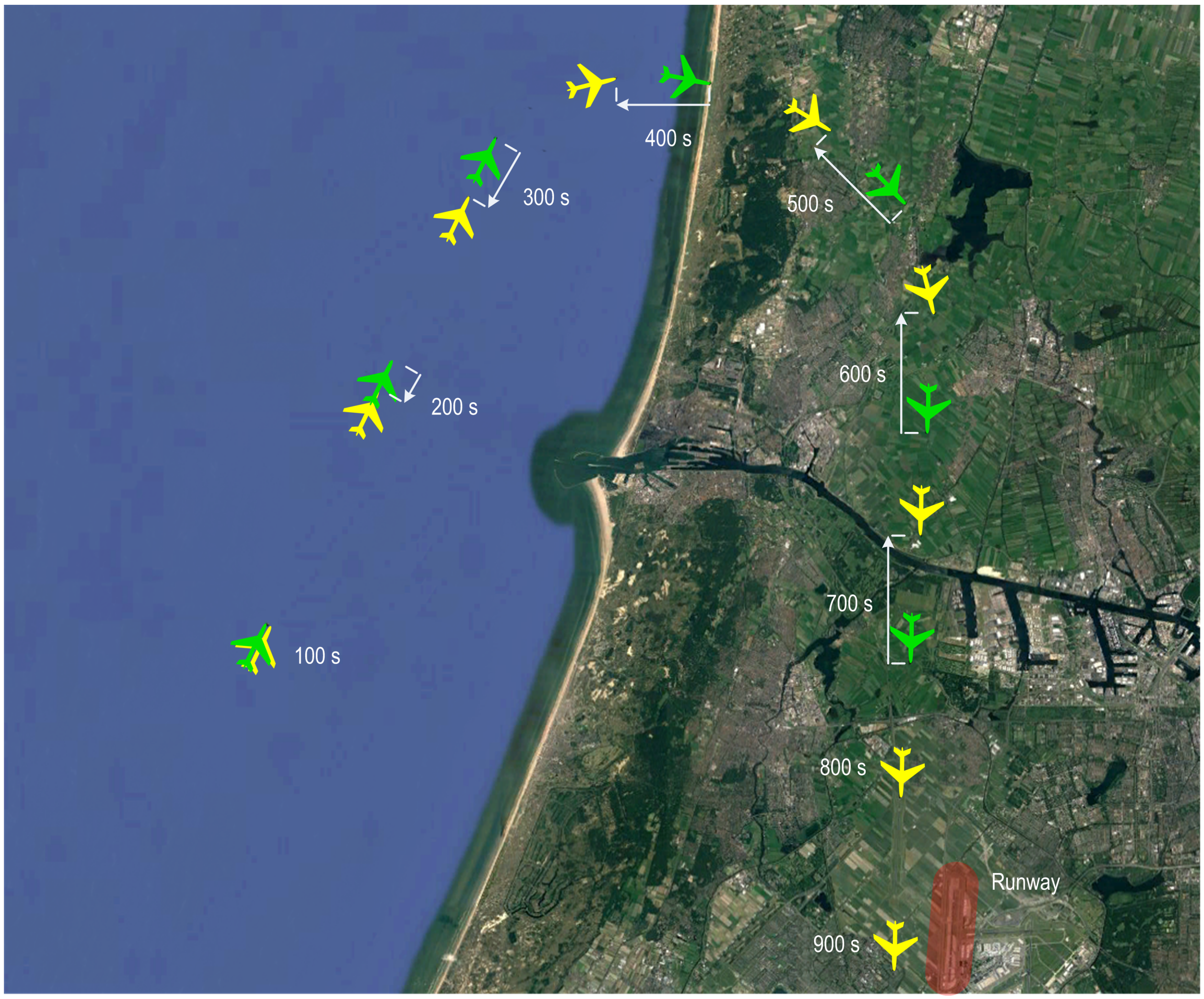

Fig. 5 Comparison of normal (green) and modified (yellow) aircraft 3 location throughout simulation.

different time points. The yellow markers indicate the modified position of aircraft 3 produced through the iterative application of our method at comparable points in time. The following items help provide context to points shown in Fig. 5:

1) At 820 seconds, the first aircraft has already landed (which happened at 687.7s). The second and third aircraft are maneuvering towards the runway.
2) At 900 seconds, the third aircraft is now far downrange from the runway and has not landed. The second aircraft previously landed at 824.4s.
3) At 950 seconds, the third aircraft continues to travel further downrange from the runway in a go-around.

If we compare the distances between the original run of the scenario and this modified version for aircraft 3 (Fig. 6), we notice that the distance grows particularly large as time continues. Note the sharp increases shown in Fig. 6 between 824 seconds, when the original aircraft lands, and 900 seconds, when the simulation is terminated. This is indicative of the aircraft throttling up and initiating a go-around maneuver.

B. FA3FA3

The FA3FA3 scenario was iteratively run through our method up to the 100 second simulation time limit. In this situation, the delays in actions only increased the distances between the aircraft by a few hundred meters. Specifically, Fig. 7 shows the largest observed difference between any aircraft's original and modified simulation positions was

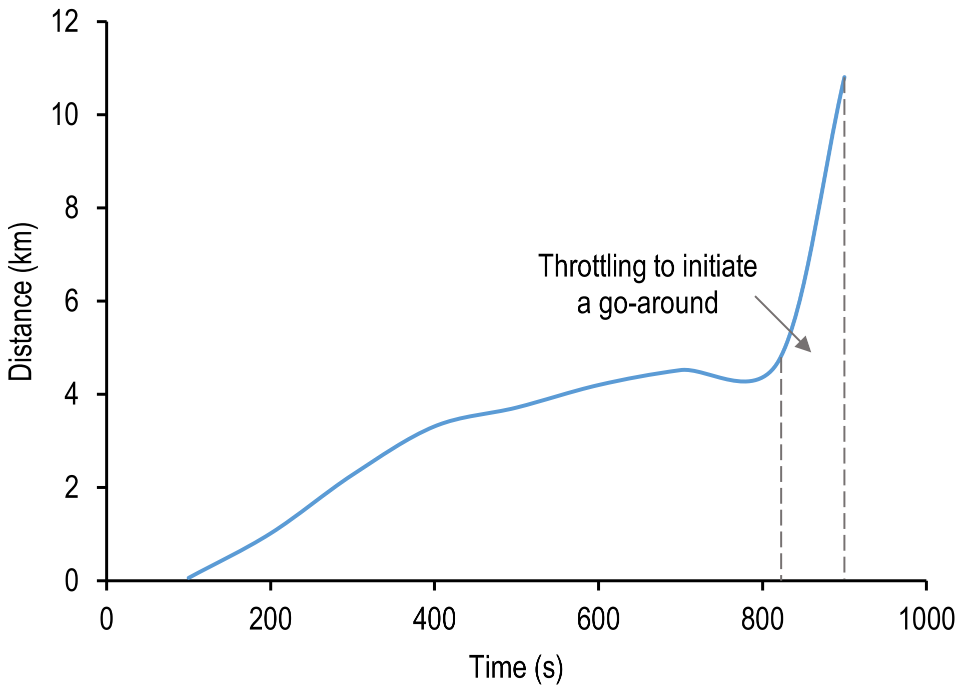

Fig. 6 Plot of the distance between aircraft 3 (TAC1) from the original, unmodified FA1FA1 scenario and the same aircraft in the modified scenario.

0.37km. These delays ultimately had little effect on the trajectories of each aircraft and all three aircraft were able to land successfully.

VI. Discussion and Conclusions

In this work we described a novel method for synergistically using formal verification and simulation to explore the space around simulated air traffic scenarios, which overcomes the some of limitations of the two methods independently. This method provides the kind of analysis needed and not available early in the investigation of a concept of operation. This analysis has given us results that are compelling both for their implications for emerging air traffic management concepts of operation and model based analyses. Both of these, and implications for future work, are discussed below.

Our results highlight the need for proper investigation of potential flight scenarios and the dangers of overloaded ATC systems. In order to effectively use the airspace and ensure that planes can execute their proper landing procedures, the design of concepts of operation need to ensure an allocation of tasks that maintains safety in airspace. The proper function allocation can allow for even extremely overloaded agents to execute landing procedures when this burden is spread across all agents as shown by FA3FA3.

A. Air Traffic Management

We hypothesize that the delays in actions that would result from producing scenarios through the application of our method would reduce the space between the simulated aircraft. Further, we hypothesize that this effect would have the most impact on the FA1FA1 scenario due to the observed function allocation imbalance putting the majority of authority and responsibility on the air traffic controller.

Our first hypothesis proved to be incorrect: increasing the amount of time between actions actually increased the space between aircraft. However, our second hypothesis did prove to be true: the effect of the action delays were more pronounced in the FA1FA1 scenario. In fact, the failed approach observed in our modified version of this scenario is very concerning because in NextGen air traffic management concepts, airspace efficiency is critical to successful operation. A go-around due to operational conditions would result in unexpected delays that could propogate throughout the air transportation system.

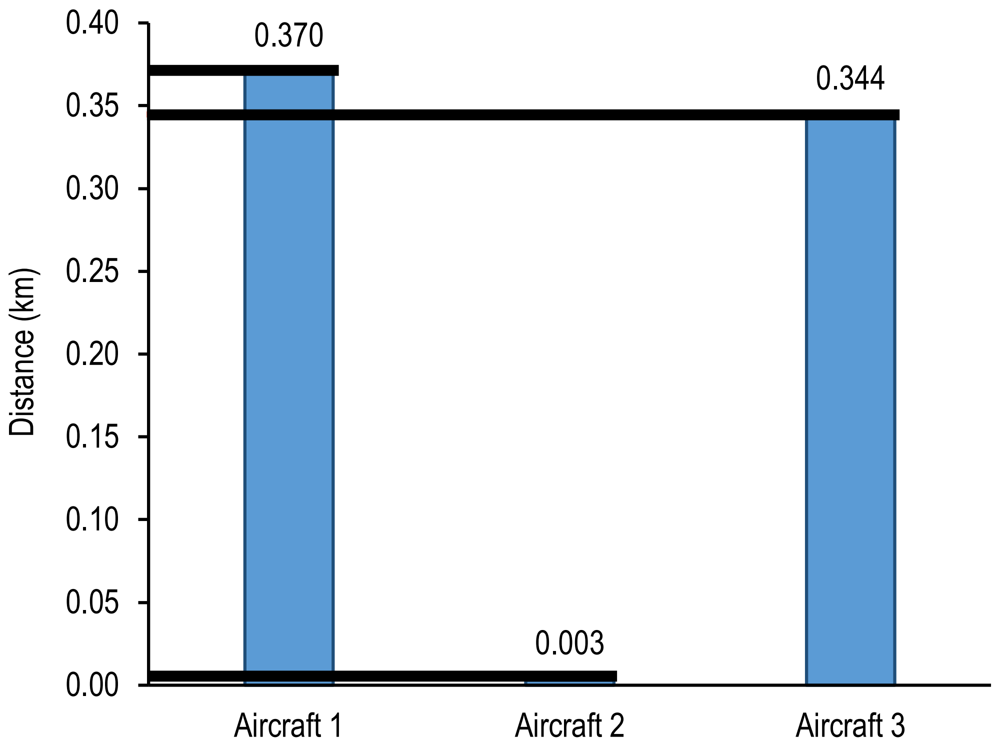

Fig. 7 The largest difference between aircraft positions in the original and modified (to 100 seconds) FA3FA3 scenarios.

These are compelling results for air traffic management considerations. First, they highlight the critical role function allocation plays in emerging air traffic management concepts of operation. The balanced function allocation from FA3FA3 was able to cope with the timing parameters discovered for avoiding human operator overload. However, the comparatively biased FA1FA1 dramatically failed in this respect, resulting in a situation where an aircraft had to break off the approach. Thus, this analysis provides compelling evidence that balanced function allocations have the potential to be more robust and efficient than imbalanced ones.

Second, our results illustrate the importance of accounting for human timing and work limitations when considering concepts of operation. The original analyses of FA1FA1 and FA3FA3 [3, 21] measured the total amount of taskwork associated with the two function allocations. However, they did not account for the timing of actions, nondeterminism introduced by action prioritization, limits on the number of actions humans can execute at a time, or total limits on human operator taskload. Under these assumptions, no major problems were observed in the scenarios. However, by showing how accounting for these in the presented analyses could result in dramatic delays, these factors should be accounted for in future analyses. We can conclude the important of pre-planning and how errors early on can propagate largely much later on in simulation time and therefore, real life scenarios as well.

B. Formal Verification and Simulation

In the results presented in this paper, we successfully used formal verification with simulation together to analyze an industrial scale aerospace system. This is a major breakthrough for several reasons.

First, the other dual analytic approaches have either focused on limited aspects of a simulation [24–27] or on only model checking the behavior in a simulation trace [28–30]. Thus, this works demonstrates the feasibility of our approach and demonstrates its applicability by providing realistic results.

Second, the complexity of analyzed scenarios demonstrates the ability of our approach to scale to industrial-sized problems. This is significant given that formal verification is traditionally limited by the complexity of the analyzed model [13]. The fact that we were able to successfully deploy our method to these scenarios suggests that many more complex conditions could be evaluated with our method. This could enable formal verification to be used on systems previously too complex for meaningful analysis. This could have profound implications for the reliability and safety of complex systems.

C. Direction for Future Work

There are some limitations to our approach that could be addressed in future work.

First, the action timings included in the analyses were derived from expert opinion. In reality there will likely be variability in the time needed to perform actions. Our method [16] currently supports the ability to account for variance in action time execution. Future work should investigate how to incorporate action timings that realistically account for this variance.

Second, the work presented here does not allow for variations in the actions that are performed due to human error, human-human communication, or changes in the environmental conditions. These could also impact taskload by introducing additional actions or time constraints. Work within the extended formal methods literature has focused on how human error, anomalous system conditions, and miscommunications can be generated in formal models so that verifications can assess their impact on system performance [22, 34–40]. Future work should investigate how these approaches could be adapted for use in our method.

Finally, the method was only applied to two air traffic management scenarios. There are likely many other emerging concepts that could produce unexpected effects as a result of the limitations on human operator taskload. Future work should investigate how our method could be applied to other emerging air traffic management concepts of operation.

Acknowledgments

This work was supported by the grant "Scenario-Based Verification and Validation of Autonomy and Authority" from the NASA Ames Research Center under award number NNX13AB71A. The authors would like to thank the developers of WMC and AARStudies.

References

[1] Feigh, K., and Pritchett, A., "Function Allocation between Human and Automation and Between Air and Ground," *Encyclopedia of Aerospace Engineering*, 2016.

[2] Lewis, T. A., Aweiss, A. S., Guerreiro, N. M., and Daiker, R. J., "A Review of Function Allocation and En Route Separation Assurance," Tech. Rep. NASA/TM 2016 6219343, Langley Research Center, Hampton, Virginia, 2016.

[3] Pritchett, A. R., Bhattacharyya, R. P., and IJtsma, M., "Computational Assessment of Authority and Responsibility in Air Traffic Concepts of Operation," *Journal of Air Transportation*, Vol. 24, No. 3, 2016, pp. 93–101.

[4] Tomiyama, T., "Function Allocation Theory for Creative Design," *Procedia CIRP*, Vol. 50, 2016, pp. 210–215.

[5] Pritchett, A., and Feigh, K., "Simulating first-principles models of situated human performance," *Proceedings of the IEEE First International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support*, IEEE, Piscataway, 2011, pp. 144–151.

[6] Pritchett, A. R., Kim, S. Y., Kannan, S. K., and Feigh, K., "Simulating situated work," *2011 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA)*, 2011, pp. 66–73.

[7] Popescu, V., Clarke, J., Feigh, K. M., and Feron, E., "ATC Taskload Inherent to the Geometry of Stochastic 4-D Trajectory Flows with Flight Technical Errors," *CoRR*, Vol. abs/1102.1660, 2011.

[8] Metzger, U., and Parasuraman, R., "Automation in future air traffic management: Effects of decision aid reliability on controller performance and mental workload," *Human Factors*, Vol. 47, No. 1, 2005, pp. 35–49.

[9] Bhattacharyya, R. P., and Pritchett, A., "Synthesis of allocations of authority in air traffic concepts of operation," *IEEE/AIAA 35th Digital Avionics Systems Conference*, IEEE, 2016.

[10] Martin, L., Bienert, N., Claudatos, L., Gujral, V., Kraut, J., and Mercer, J., "Effects of task allocation on air traffic management human-automation system performance," *IEEE/AIAA 35th Digital Avionics Systems Conference*, IEEE, 2016.

[11] Pritchett, A. R., Kim, S. Y., and Feigh, K. M., "Measuring human-automation function allocation," *Journal of Cognitive Engineering and Decision Making*, Vol. 8, No. 1, 2014, pp. 52–77.

[12] Wing, J. M., “A specifier’s introduction to formal methods,” *Computer*, Vol. 23, No. 9, 1990, pp. 8, 10–22, 24.

[13] Clarke, E. M., Grumberg, O., and Peled, D. A., *Model checking*, MIT Press, Cambridge, 1999.

[14] Mansouri-Samani, M., Pasareanu, C. S., Penix, J. J., Mehlitz, P. C., O’Malley, O., Visser, W., Brat, G., Markosian, L., and Pressburger, T., “Program Model Checking: A Practitioner’s Guide,” Tech. rep., Intelligent Systems Division, NASA Ames Research Center, Moffett Field, 2007.

[15] Houser, A., Ma, L. M., Feigh, K., and Bolton, M. L., “A formal approach to modeling and analyzing human taskload in simulated air traffic scenarios,” *2015 International Conference on Complex Systems Engineering*, 2015, pp. 1–6.

[16] Houser, A., Ma, L. M., Feigh, K., and Bolton, M. L., “Using Formal Methods to Reason About Taskload and Resource Conflicts in Simulated Air Traffic Scenarios,” *Innovations in Systems and Software Engineering*, ND, p. 15 pages. In Press.

[17] Pritchett, A. R., “Simulation to Assess Safety in Complex Work Environments,” *The Oxford handbook of cognitive engineering*, edited by J. D. Lee and A. Kirlik, Oxford University Press, New York, 2013, Chap. 22, pp. 352–366.

[18] Pritchett, A. R., Feigh, K. M., Kim, S. Y., and Kannan, S. K., “Work Models that Compute to Describe Multiagent Concepts of Operation: Part 1,” *Journal of Aerospace Information Systems*, Vol. 11, No. 10, 2014, pp. 610–622.

[19] Gelman, G. E., “Comparison of Model Checking and Simulation to Examine Aircraft System Behavior,” Ph.D. thesis, Georgia Institute of Technology, 2012.

[20] Pritchett, A. R., Kim, S. Y., and Feigh, K. M., “Modeling human–automation function allocation,” *Journal of Cognitive Engineering and Decision Making*, Vol. 8, No. 1, 2014, pp. 33–51.

[21] IJtsma, M., Pritchett, A. R., and Bhattacharyya, R. P., “Computational Simulation of Authority-Responsibility Mismatches in Air-Ground Function Allocation,” *Proceedings of the 18th International Symposium on Aviation Psychology*, Write State University, Dayton, 2015. 6 pages.

[22] Bolton, M. L., Bass, E. J., and Siminiceanu, R. I., “Using formal verification to evaluate human-automation interaction in safety critical systems, a review.” *IEEE Transactions on Systems, Man and Cybernetics: Systems*, Vol. 43, No. 3, 2013, pp. 488–503.

[23] Weyers, B., Bowen, J., Dix, A., and Palanque, P. (eds.), *The Handbook of Formal Methods in Human-Computer Interaction*, Springer, Cham, 2017.

[24] Hu, A. J., “Simulation vs. formal: Absorb what is useful; reject what is useless,” *Proceedings of the Third International Haifa Verification Conference*, Springer, Berlin, 2008, pp. 1–7.

[25] Yuan, J., Shen, J., Abraham, J., and Aziz, A., “On combining formal and informal verification,” *Computer Aided Verification*, Springer, 1997, pp. 376–387.

[26] Gelman, G., Feigh, K. M., and Rushby, J., “Example of a Complementary use of Model Checking and Agent-based Simulation,” *IEEE International Conference of Systems Man and Cybernetics*, IEEE, Piscataway, 2013, pp. 900–905.

[27] Gelman, G., Feigh, K., and Rushby, J., “Example of a Complementary use of Model Checking and Human Performance Simulation,” *IEEE Transactions on Human-Machine Systems*, Vol. 44, No. 5, 2014, pp. 576–590.

[28] Yasmeen, A., Feigh, K. M., Gelman, G., and Gunter, E. L., “Formal analysis of safety-critical system simulations,” *Proceedings of the 2nd International Conference on Application and Theory of Automation in Command and Control Systems*, IRIT Press, 2012, pp. 71–81.

[29] Stuart, D. A., Brockmeyer, M., Mok, A. K., and Jahanian, F., “Simulation-verification: Biting at the state explosion problem,” *IEEE Transactions on Software Engineering*, Vol. 27, No. 7, 2001, pp. 599–617.

[30] Chen, X., Hsieh, H., Balarin, F., and Watanabe, Y., “Automatic trace analysis for logic of constraints,” *Proceedings of the Design Automation Conference*, IEEE, 2003, pp. 460–465.

[31] Emerson, E. A., “Temporal and modal logic,” *Handbook of Theoretical Computer Science*, edited by J. van Leeuwen, A. R. Meyer, M. Nivat, M. Paterson, and D. Perrin, MIT Press, Cambridge, 1990, Chap. 16, pp. 995–1072.

[32] De Moura, L., Owre, S., Rueß, H., Rushby, J., Shankar, N., Sorea, M., and Tiwari, A., “SAL 2,” *International Conference on Computer Aided Verification*, Springer, 2004, pp. 496–500.

[33] Miller, G. A., “The magical number seven, plus or minus two: Some limits on our capacity for processing information,” *Psychological Review*, Vol. 63, No. 2, 1956, p. 81.

[34] Bolton, M. L., and Bass, E. J., “Using relative position and temporal judgments to identify biases in spatial awareness for synthetic vision systems,” *The International Journal of Aviation Psychology*, Vol. 18, No. 2, 2008, pp. 1050–8414.

[35] Bolton, M. L., Bass, E. J., and Siminiceanu, R. I., “Generating phenotypical erroneous human behavior to evaluate human-automation interaction using model checking,” *International Journal of Human-Computer Studies*, Vol. 70, No. 11, 2012, pp. 888–906.

[36] Bolton, M. L., and Bass, E. J., “Comparing perceptual judgment and subjective measures of spatial awareness,” *Applied Ergonomics*, Vol. 40, No. 4, 2009, pp. 597–607.

[37] Bolton, M. L., and Bass, E. J., “Generating erroneous human behavior from strategic knowledge in task models and evaluating its impact on system safety with model checking,” *IEEE Transactions on Systems, Man and Cybernetics: Systems*, Vol. 43, No. 6, 2013, pp. 1314–1327.

[38] Pan, D., and Bolton, M. L., “Properties for formally assessing the performance level of human-human collaborative procedures with miscommunications and erroneous human behavior,” *International Journal of Industrial Ergonomics*, ND. doi:http://dx.doi.org/10.1016/j.ergon.2016.04.001.

[39] Bolton, M. L., “Model checking human–human communication protocols using task models and miscommunication generation,” *Journal of Aerospace Information Systems*, Vol. 12, 2015, pp. 476–489.

[40] Bolton, M. L., “A task-based taxonomy of erroneous human behavior,” *International Journal of Human-Computer Studies*, Vol. 108, 2017, pp. 105–121.