REAL-TIME SIMULATION OF A DISTRIBUTED CONFLICT RESOLUTION ALGORITHM

Graham T. Spence* and David J. Allerton*
Richard Baumeister and Regina Estkowski****
***Department of Automatic Control and Systems Engineering, University of Sheffield, UK**
****The Boeing Company, Seattle, US**

Keywords: *aircraft, separation, resolution, distributed, automated*

Abstract

This paper describes a real-time simulation study of conflict resolution in air traffic management where aircraft regularly broadcast their position, heading and speed and the computation of conflict resolution can be either distributed among the aircraft or computed at a single location and broadcast to all aircraft on the network. Communication latency is included in the simulation by two methods. The first is based on TCP/IP message latency over the Internet, while the second uses Iridium satellite data communications, which inherently provide relatively high latencies and communication errors. The architecture of the simulator and the real-time conflict resolution algorithms are described, together with results from simulation studies.

1 Introduction

Modernization plans by the FAA and the EU are shifting from centralized decision making by an aircraft controller communicating by voice towards decentralized conflict resolution where messages are broadcast regularly by aircraft [1].

In this environment, each aircraft within a region has knowledge of the position, speed and heading of other aircraft from monitoring of messages, typically broadcast once per second by each aircraft. In addition, each aircraft will have sufficient computing power to resolve conflicts and communicate these resolutions to other aircraft or to accept resolutions by other members of the network and then monitor the situation to provide sufficient situational awareness for flight crew. In effect, a local conflict resolver provides additional inputs to flight management systems, in order to modify the flight plan to ensure safe conflict-free trajectories.

However, such systems can introduce significant problems. Firstly, the sensor accuracy is bounded in terms of measurements of position, altitude, heading and speed. Secondly, latency in acquiring access to the network to transmit messages can vary considerably, resulting in decision-making based on out-of-date reports. Thirdly, the effect of wind introduces uncertainties in the prediction of future safe trajectories. Finally, the sampling rate of data within the network is limited by the bandwidth of the communication system. For example, for a data transmission rate of 1 Hz in a network of aircraft, with speeds up to 300 m/s, missed or delayed measurements can result in potential position errors up to 1 km.

The objective of this paper is to describe real-time simulations of a network of aircraft autonomously determining conflict-free flight. The goal of these simulations is to demonstrate that safe aircraft separation can be maintained although the message latency of the broadcast data is variable. The paper outlines the architecture of the traffic simulator, the distributed conflict resolution algorithm and the organization to combine both simulations including transport delays in networks. The paper also presents results from studies of traffic patterns where the traffic generator and the conflict resolver are separate processes running in different countries communicating via Internet and Iridium links. These studies include traffic patterns designed to illustrate the

effectiveness of this approach and to expose conditions that result in complex conflict resolution scenarios. The results presented in the paper are based on assumptions of sensor resolution, network data rates and latencies and also on the computational processing speed of airborne systems. Indications of the research linking the traffic model executing at Phantom Works in Boeing with the separation management software running at the University of Sheffield are encouraging and show that autonomous resolution of conflicts for 50 aircraft in a 50 Km square region is feasible.

2 Simulation Architecture

An air-traffic simulator has been developed to enable a variety of aircraft separation algorithms to be developed and tested. The traffic simulator runs as a standalone process, implementing resolution commands and broadcasting aircraft telemetry [2]. The separation management software receives broadcast messages and executes the conflict resolution algorithms for each aircraft independently. With this architecture, message latency is introduced to the simulations between the traffic models and the separation software.

2.1 Communication Interface Overview

One aspect of the traffic simulator architecture is to be able to integrate the development activities of different research groups for real-time analysis. To realize this, the traffic simulator has a TCP/IP interface that allows it to communicate either through binary data messages or ASCII byte streams routed through Iridium transceivers (see Figure 1 and Section 2.2). Initial studies using TCP/IP messages across the Internet from sites in the US to the UK have indicated that network latency is significantly less than that anticipated in future aircraft networks (e.g. ADS-B [3]) if using satellite links. When the traffic simulator is communicating over the Internet a latency model can be applied to simulate delay in message reception. When using Iridium as a communication channel for several aircraft, the latency of the data communication hardware is included in the message handling.

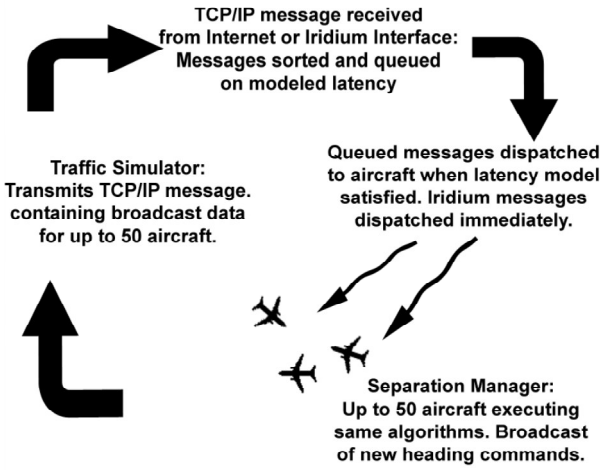

Fig. 1. Traffic model and separation manager communication architecture.

2.2 Iridium Based Communications

Algorithms and methods for automated airspace management must be able to provide safe performance when operating with data that is corrupted by non-reliable communication links. This study uses the Iridium [4,5] system as a communication link to evaluate our automated airspace management concepts.

The Iridium system is a satellite system providing near global coverage for Iridium phone and data subscribers. It consists of a constellation of low orbiting satellites and associated ground gateways. Although the present Iridium system cannot simultaneously support thousands of aerial platforms required for global ATM it provides a test-bed to evaluate automated airspace management algorithms. A dedicated Iridium data channel has an approximate bandwidth of 2.4 Kbps [5], which is close to the data rate envisaged for future systems. One of our test objectives is to measure and quantify the performance of the Iridium system when servicing simulated automated air traffic management nodes.

Software has been developed by Boeing to test these communication capabilities and to serve as the prototype communication medium and software interface for planned future flight tests. The software, named JMS/Iridium

Software (JIRID), creates an interface between a PC and an Iridium transceiver, allowing the communication parameters to be dynamically modified. JIRID provides TCP/IP connectivity providing a gateway to the traffic model or the separation manager.

JIRID supports both ad-hoc data streams and file transfers across the Iridium system. Users can control the number of records to be sent, transmit rate, and the data mode. Data mode selection enables the data to be sent through an Iridium gateway or directly to an Iridium L Band Transceiver (LBT).

Network reliability is a crucial parameter of any future airborne communications link. One measurement of the reliability of the Iridium system is the signal strength captured by the JIRID program when linked to a transceiver. This measurement approximates the link margin (dB) between the Iridium satellite in view and a user transceiver. Signal bar strength is reported as an integer between 0 and 5. A dropped call can occur because of low signal to noise ratio (signal blockage, satellite hand-off) or from frequency interference (e.g. INMARSAT). Our observations are that data calls are often dropped if the signal bar level falls below 2.

Tests of signal bar levels have been conducted at two locations. The first location (Sheffield, UK) used a transceiver with the antenna partially obscured. The summary of this test conducted on January 24th 2008 is as follows: A total of 10491 data points were collected over 18.8 hours (there are approximately 6.447 seconds between samples). There were a total of 3332 zeros over the course of the test and the probability of receiving signal bar strength greater than zero was 0.682. The second location was also based in Sheffield, UK, but with the transceiver antenna having an almost 360˚ clear sky view. Figure 2 presents the signal bar levels for this particular test conducted between the 25th and 28th of January 2008. A total of 42347 data points were collected over 75 hours. The results for both signal tests are presented in Table 1. Clearly the location of the transceiver antenna is important. An optimum sky view should provide clear line of sight for 8.2˚ elevation around 360˚ [5]. Achieving optimum clear sky views is difficult in urban areas so it is expected that similar tests executed for airborne transceivers should yield further improved signal strength measurements.

	Location 1	**Location 2**
Number of Data Points Recorded	10491	42347
Number of Zeros recorded	3332	64
Probability that signal bar value > zero	0.682	0.998

Table. 1. Properties of signal strength monitored at two sites.

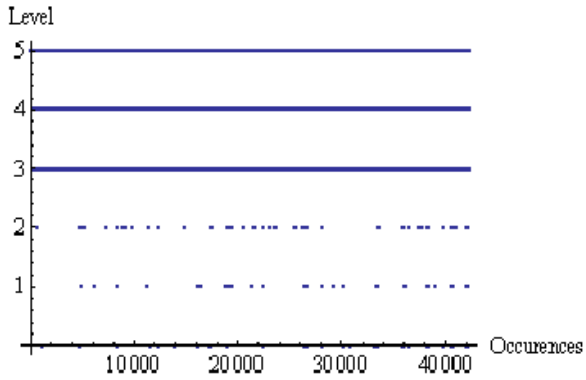

Fig. 1. Time distribution of signal bar levels at transceiver Location 2.

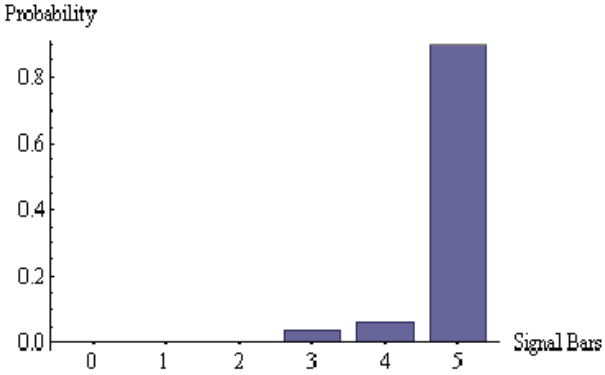

Fig. 3. Probability of signal bar levels at transceiver Location 2.

3 Conflict Resolution Algorithm

In this study the algorithms developed for conflict detection and resolution are based on assumptions that each aircraft within a 50 km by 50 km region will receive broadcast data for up to 50 aircraft containing position, altitude, heading and speed. Each aircraft follows a flight plan through the region, such that any deviation is minimal. Only heading change commands are issued by the resolution algorithm.

The design of the algorithm is based on a distributed architecture of networked aircraft. The resolution software has been developed with an object-oriented methodology. Each aircraft is treated as a software object containing the conflict detection and resolution routines and additionally the latest data for each aircraft in the traffic network. This design allows the resolution algorithms to be executed in a distributed configuration amongst the traffic network, or to be packaged together as a single program and executed at a centralized location. Note that the conflict resolution algorithm described is stateless in the sense that data from previous network transmissions is not required to compute resolutions for the current state of the aircraft network.

The following sections describe the aircraft separation algorithm working in a latency-free configuration. Modifications to the algorithm required for operation in an environment with variable message latency are subsequently described.

3.1 Conflict Detection

In this discussion the term conflict detection refers to the detection of potential separation violations without the application of any resolution actions. The conflict detection method for a pair of aircraft applied in the overall algorithm is based on the TCAS [6] alerting system using the Bramson criteria for the case with near parallel and slowly converging encounters. This technique is based on relative navigation geometry and provides a useful measure of the time to the closest point of approach, Tau (Γ), between a pair of aircraft [7]. A conflict-alerting threshold time defines a time limit to discriminate between potential conflicts that should be resolved or ignored. For these studies an alert threshold is set to a value of 60 seconds. Each aircraft determines the value of Γ for every other aircraft. To account for message latency, the detection routine estimates the current location of all other aircraft from their last known position in the previous broadcast. Using the estimated aircraft locations Γ is determined and compared with the alert threshold value. A potential collision alert is raised if Γ is less than the alert threshold (and positive) and the computed miss distance (M_{cpa}) at the closest point of approach is less than the desired miss distance (M_d) of the encounter. The current simulations use a target miss distance of 2 km.

3.2 Conflict Resolution for Aircraft Pairs

The goal of this conflict resolution algorithm is to create a heading command that will resolve the conflict with a miss distance of M_d. To determine a new commanded heading, the algorithm creates an avoidance vector, A_v, as shown in Figure 4. A_v is the sum of the vector from an aircraft's current position to the closest point of approach (CPA_v), and a vector perpendicular to the current flight path with a magnitude of $M_d/2 - M_{cpa}/2$, where M_{cpa} represents the projected miss distance between aircraft at the closest point of approach.

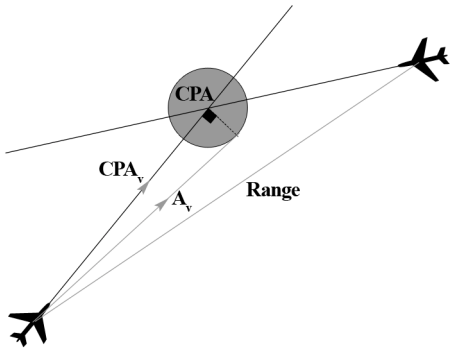

Fig. 4. The avoidance vector, A_v, steers aircraft away from the CPA.

The avoidance vector provides the basis for determining an ideal heading update (independent of aircraft dynamics). Currently, the resolution algorithm issues heading commands iteratively to provide smaller heading changes (maximum 3˚ per second) for

the traffic simulator. For aircraft to complete a resolution maneuver, the solution executes over many iterations. The algorithm can run in either a distributed or centralized form. Both aircraft in the pair execute the same equations independently and generate heading changes as outlined. Because both aircraft compute their own heading changes, the conflict resolution is cooperative in the sense that both aircraft maneuver in order to generate the desired miss distance.

3.3 Conflict Resolution for Multiple Aircraft

Although pair-wise aircraft resolutions provide good test cases, the goal is to manage aircraft separation in densely populated airspace. In such environments, conflict resolutions limited to pair-wise conflicts of many aircraft may not provide safe avoidance maneuvers. This section details how the conflict resolver is extended to *n* aircraft.

A self-organizing network of aircraft is formed by using multiple avoidance vectors for a single aircraft resolution (one vector for each *n* aircraft that raises a potential collision alert). By summing all of the contributing avoidance vectors to determine the resultant heading change, the resolver accounts for multiple potential collisions during each conflict resolution computation. For each aircraft, a conflict detection check is executed against all other known aircraft. Only those aircraft that raise an alert, for a particular aircraft *i*, contribute to the final avoidance vector, $\mathbf{A}_i$, of an aircraft (Equation 1).

$$\mathbf{A}_i = \sum_{\substack{k=0,\\ k!=i}}^{k=n-1} \left(\mathbf{P}_{fvk} + \mathbf{CPA}_{vk}\right) w_k \qquad (1)$$

Where $\mathbf{P}_{fvk}$ is the vector perpendicular to the flight vector of aircraft *i* and ω_k is a weighting applied to each contributing avoidance vector and inversely proportional to the value of Γ for the conflicting aircraft. If the value of $|\mathbf{A_i}|$ is greater than zero then the conflict resolution algorithm issues a heading change command to the host aircraft. Otherwise aircraft steering is determined by a proportional navigation system that tracks the aircraft to its next waypoint.

If all aircraft in the network are executing the same resolution algorithm, cooperative behavior emerges and the aircraft network becomes self-organizing.

3.4 Incorporating Rules of the Air

Initially the resolver consisted of a rule that commands aircraft to turn right when resolutions are required. Such simplistic behavior fails to provide an adequate conflict resolution solution. Testing has shown that resolution instabilities, in the form of aircraft heading oscillations, can occur when the conflict geometry between two aircraft has a narrow incident angle. Other research groups have also demonstrated similar oscillations in collision avoidance algorithms based on Artificial Potential Field techniques [8,9,10,11]. These instabilities can be brief or last a significant length of time depending on the initial encounter geometry. Such heading oscillations are unhelpful for a human pilot, increasing both the pilot workload and the distance traveled to resolve any conflicts.

To overcome this oscillatory behavior, some *rules of the air* have been included in the determination of the conflict response. Most conflicts can be resolved using a right turn avoidance vector as shown in Figures 5(a) and 5(b). As the incidence angle between conflicting aircraft reduces, the right turn rule is less robust and is the major cause of oscillatory behavior. If both aircraft turn right, an immediate collision may be avoided as Γ increases beyond the alert threshold. However, both aircraft fail to cross flight tracks. Subsequent attempts to alter course back to track will raise a new alert (the initiation of an instability), resulting in both aircraft turning right; again failing to cross tracks. This algorithm avoids such a situation by requiring slower aircraft to alter course, while allowing faster aircraft to maintain track. When the incidence angle between two conflicting aircraft is below 22.5˚, the turn right rule is replaced by a 60˚ track deviation rule, as is shown in Figure 5(c). The magnitude of the deviation is calculated so that upon return to the original track, the diverted aircraft is behind the faster aircraft by at least the target miss distance. For

example, if a slower aircraft is to the left of the conflict geometry a -60˚ heading change is commanded. This technique provides a predicable outcome to conflict resolution in the absence of speed and altitude changes. An added benefit is that using such a track deviation also permits overtaking on collinear aircraft tracks.

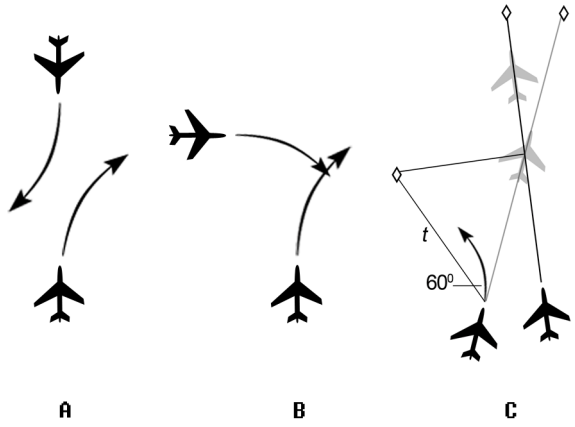

Fig. 5. Geometries A (head-on) and B (side-on) are resolved using a *turn right* rule. In C, slow aircraft deviate from track.

4 Separation Management Simulations

The testing of aircraft separation algorithms can be executed locally, or on a LAN, over the Internet or using Iridium. When using the Internet, latency is negligible and assessing the performance of the separation algorithms requires broadcast messages to be artificially delayed. Rather than delaying messages, potential collision simulations have been executed over both the Internet and Iridium. The effect of message latency on the resolution algorithms over both communication mediums can be observed.

Although many potential collision scenarios have been previously simulated, this paper presents four scenarios that attempt to stress different facets of the separation manager. In each case all aircraft remain at the same altitude, their ground speed is fixed and only heading commands issued. This constrains the aircraft separation problem by removing alternative solutions, for example altitude changes.

Case 1 consists of 8 aircraft prearranged to converge on a point simultaneously. Each flight plan is programmed for a linear track through the coincident point. If new heading commands are issued by the separation manager and an aircraft moves off-track, waypoint navigation software will return an aircraft back on-track after the resolution is complete.

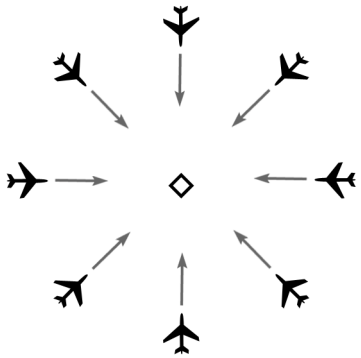

Fig. 6. Eight aircraft converging on a coincident point (Case 1).

Case 2 is similar to Case 1 in that 5 aircraft have flight plans that take them through a coincident point simultaneously. Additionally, each aircraft follows a non-linear flight plan that should eventually return to its starting location, invoking a potential collision state for a second time. Testing with curved flight plans also stresses the separation manager by magnifying aircraft position estimation errors in the resolution algorithm.

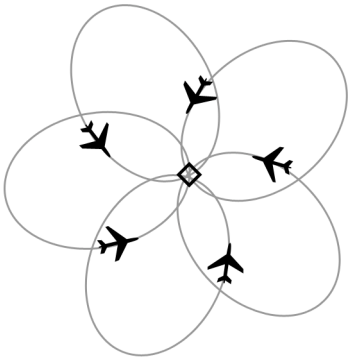

Fig. 7. Five aircraft with non-linear trajectories (Case 2).

Case 3 involves 13 aircraft initially on parallel tracks heading west. The lateral spacing

between these aircraft is approximately 5.5 km (3 nm). A single aircraft is on a track heading east with a potential head-on collision with the middle aircraft in the 'wall'. This test highlights the cooperative nature of the separation manager and exposes any associated knock-on effects, where aircraft are initially closely spaced.

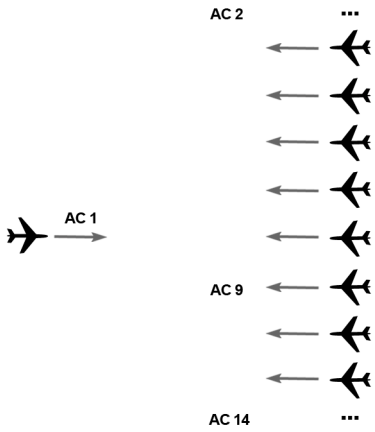

Fig. 8. Single aircraft heading towards a wall of 13 aircraft (Case 3).

Case 4 consisted of a set of 50 randomly positioned aircraft with random headings and ground speeds (within a suitably fixed range). All aircraft are set on a linear track matching their initially generated headings. In the generation of the initial aircraft locations, only aircraft within a 50km by 50km airspace are accepted. This test is designed to stress the separation manager when operating as a centralized unit and to create unexpected conflict scenarios.

5 Separation Results

For Case 1, *one-way* message latency measurements are presented for the Iridium communication configuration. Figure 9(a) shows the recorded latency for a message containing the data for a single aircraft. Figure 9(b) shows the effect on message latency when a single message contains the data for 8 aircraft. In practice 8 separate channels (one channel per aircraft) would be used. However, it is of interest to observe how the resolution algorithms perform due to the increased loading and latency. Both diagrams report the mean and standard deviation of the message latency. In contrast, when using TCP/IP over the Internet, the values for the mean and standard deviation of message latency were 0.34 seconds and 0.055 seconds respectively. Latencies using the Internet were observed to be close to these values for packets containing up to 50 aircraft in a single message.

Figure 10(a) shows the resultant trajectories of Case 1, with all aircraft converging on a coincident point. The miss distances between the aircraft over the course of the simulation are plotted in Figure 10(b). The closest distance between any two aircraft in this case was 1.89 km.

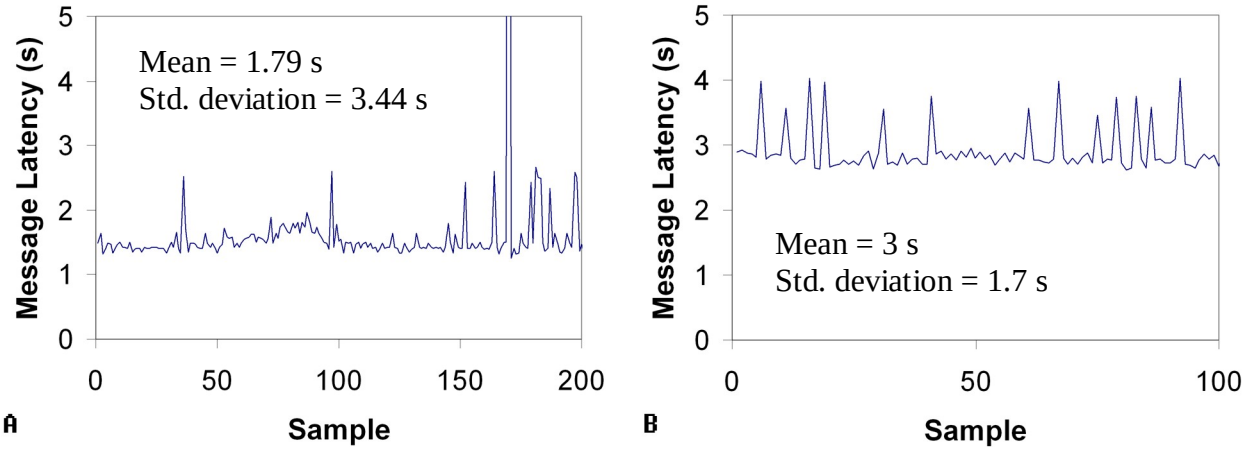

Fig. 9. (a) Latency for messages with a payload containing data for 1 aircraft. (b) Latency for messages with a payload containing data for 8 aircraft.

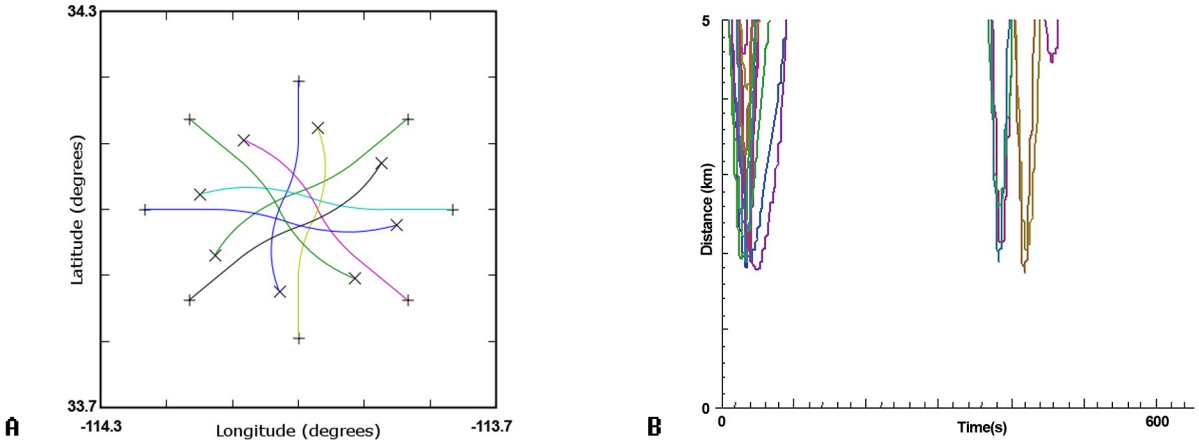

Fig. 10. (a) Case 1 trajectories at 250 seconds. (b) Mutual distances between aircraft.

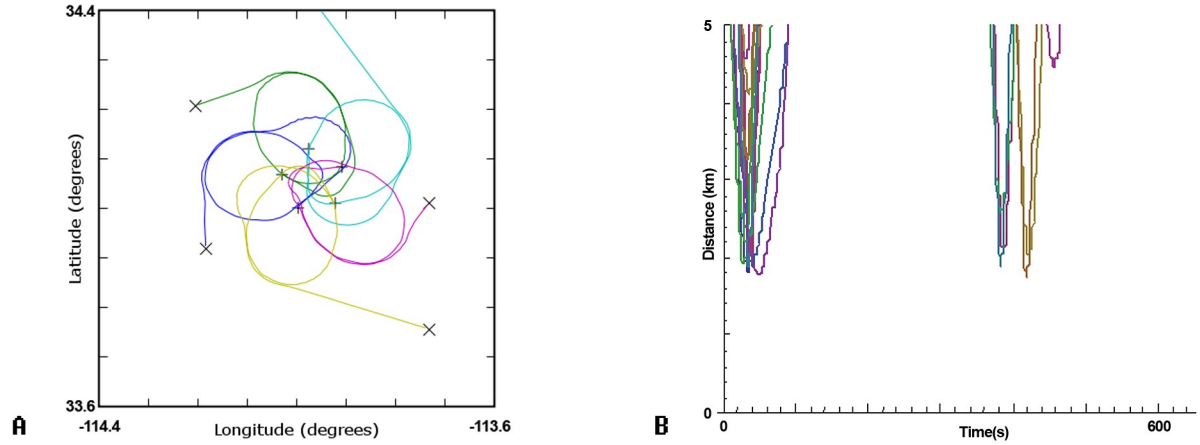

Fig. 11. (a) Case 2 trajectories. (b) Mutual distances between aircraft.

The actual trajectories and the respective mutual aircraft distances for Case 2 are presented in Figure 11. This test was executed to observe the separation manager running both with an Iridium link, and also with curvilinear trajectories containing multiple waypoints. Each circular flight consists of 21 waypoints so the results show that the track navigation and resolution algorithms operated together well. This result also highlights that, with significantly large and variable latencies, the separation manager was able to manage the airspace reasonably well. Figure 11(b) supports this by showing that all aircraft in conflict maintained an approximate minimum miss distance of 2 km. Although this result is encouraging, the position extrapolation routines used in the resolution algorithms might be affected if higher latency messages were encountered or if aircraft were permitted a higher turn-rate than 3° per second.

Results for Case 3 are presented in Figure 12. Case 3 was executed using the Internet in order to simulate higher numbers of aircraft nodes. In order to ensure safe separation between all aircraft, an ensemble maneuver is required by the wall to maintain separation distance from aircraft 1 and adjacent members of the wall. As shown in Figure 12(a), the conflict resolution algorithm moves aircraft 2-9 *up* while moving aircraft 1 *down*, leaving aircraft trajectories 10–14 unperturbed. This maneuver minimizes the total perturbation to the entire set of trajectories.

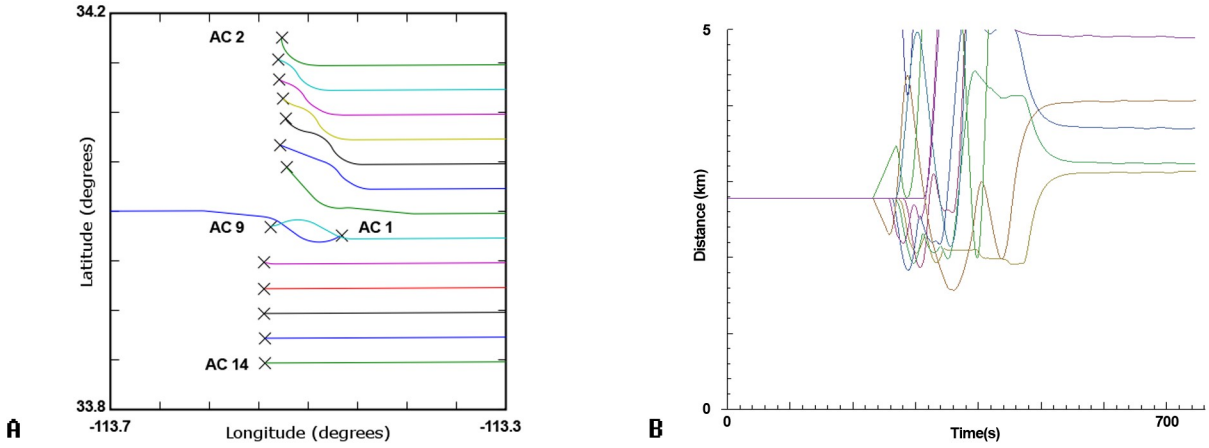

Fig. 12. (a) Case 3 trajectories. (b) Mutual distances between aircraft.

Note that the automated solution in this scenario results in dynamic maneuvers that might involve high pilot workload and crew cooperation.

The final test, Case 4, also used the Internet as its communication medium. The results of this test attempt to highlight if the conflict resolution algorithm can work in dense airspace involving no prearranged scenarios. Figure 13 plots the tracks of 50 aircraft after 700 seconds of simulation. In this case alone there are over 20 track deviations and several aircraft are involved in multiple potential collisions. Analysis of the mutual distances between all aircraft revealed two individual situations where the distance between a pair of aircraft fell just below 1 km. Multiple runs of this case produce largely similar results, with differences occurring due to the variation in latency during each test. Testing the conflict resolution algorithms with similar cases usually presents situations in which aircraft undergoing resolutions introduce new potential conflicts with aircraft that initially appear to be in conflict with no other aircraft. In many dense airspace situations these cascading collision scenarios are unavoidable. Random tests are important for adaptive conflict resolution algorithms similar to this (including Artificial Potential Methods). They provide the most direct way to create unintuitive cascading collision scenarios that can be reused to analyze the behavior of the cooperative conflict resolution.

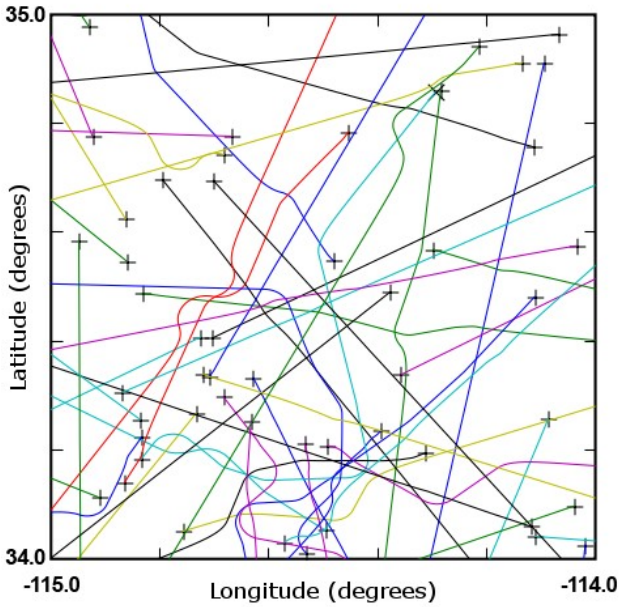

Fig. 13. Trajectories for 50 randomly placed aircraft after 700 seconds.

The Iridium test cases have shown that the straightforward position estimation works well for the latencies and turn rate restrictions encountered in these tests. The simulation has shown that it can tolerate occasional and relatively short communication drop-outs. For larger drop-outs the current implementation is less robust due to the conflict resolution algorithm relying on the latest data broadcast for the estimation of neighboring aircraft positions.

6 Future Studies

The traffic model employed in these simulations used only a basic aircraft model. For future tests we intend to use the flight simulator [12] at the

University of Sheffield as an aircraft model. The flight simulator will also be used as a research platform to enable situation awareness displays to be developed and analyzed. The intention of these displays is to provide flight crews with timely information to understand underlying conflict resolution decisions.

Other research activities include investigating aircraft network fault tolerance to provide safe separation commands during relatively long periods of communication loss.

7 Conclusion

This paper has outlined a simulation architecture used as a test platform for potential separation management using the Iridium system. Although it is unlikely that Iridium will be used in future airborne systems, it does provide satellite-based communication with variable message latency and connectivity. A candidate conflict resolution algorithm has been described which can manage densely packed airspace with a high degree of confidence. The algorithm is a combination of existing relative navigation calculations for the determination of conflict alerts, combined with a vector summation approach to yield multiple aircraft avoidance in a cooperative way. The emergent organizational behavior of the aircraft network attempts to be consistent and predictable.

The algorithm is suitable for use in a distributed network of traffic but can be adapted for centralized use in an automatic aircraft control center. It is straightforward to implement and would run on embedded microprocessors.

In summary, conflict resolution simulations using Iridium transceivers have been largely successful. It is recognized that designing and integrating distributed fault tolerance is essential if the frequency and magnitude of communication drop-outs is significant.

References

[1] RTCA Inc. *Free Flight. Report of the RTCA Board of Directors Select Committee.* Radio Technical, Commission of Aeronautics. USA, January 1995.

[2] ICAO. *Automatic Dependent Surveillance. Circular 226-AN/135.* International Civil Aviation Organization, 1983.

[3] RTCA Inc. *Minimum Operational Performance Standards for Airborne Automatic Dependent Surveillance (ADS) Equipment*, DO-212, 1992.

[4] Peterson K.M. The Iridium Low Earth Orbit Communications System. *Sarnoff Symposium*, IEEE Princeton/Central Jersey, pp 13-19, 1994.

[5] Fossa C.E, Raines R.A, Gunsch G.H, Temple M.A. An overview of the IRIDIUM (R) low Earth orbit (LEO) satellite system. IEEE National Aerospace and Electronics Conference, pp 152-159, 1998.

[6] Harman W.H. TCAS: A system for preventing midair collisions. *Lincoln Laboratory Journal*, Vol. 2, No. 3, 1989.

[7] Kayton F and Fried W.R. *Avionics Navigation Systems.* 2nd edition. John Wiley and Sons, Inc., 1997.

[8] Eby M and Kelly W. *Free Flight Separation Assurance Using Distributed Algorithms*, IEEE 1999 Aerospace Conference, March 14-18, 1999.

[9] Kelly W and Eby M. *Advances In Force Field Conflict Resolution Algorithms*, AIAA Guidance, Navigation, and Controls Conference, Paper 2000-4360, Denver, CO, August 14-17, 2000.

[10] Lee s and Park J. *Cellular Robotic Collision Free Path Planning.* 5th International Conference on Advanced Robotics, Vol. 1, pp 539-544, 1991.

[11] Khatib O. *Real-time Obstacle Avoidance for Manipulators and Mobile Robots*, International Journal of Robotics Research, Vol. 5, No. 1, pp 90-98, 1986.

[12] Allerton D. J. A. Distributed Approach to the Design of a Real-time Engineering Flight Simulator, 21st ICAS Congress, Sept. 1998.

Copyright Statement