

IMPROVED CONFLICT DETECTION FOR REDUCING OPERATIONAL ERRORS IN AIR TRAFFIC CONTROL

Russell A. Paielli* and Heinz Erzberger†
NASA Ames Research Center

Abstract— An operational error is an incident in which an air traffic controller fails to prevent two aircraft from getting closer together than the minimum required separation. The rates of such errors in the U.S. have increased significantly over the past few years. This paper presents a new conflict detection tool called TSAFE (Tactical Separation Assisted Flight Environment) that could eventually replace Conflict Alert, the legacy software that is intended to detect and warn controllers of imminent conflicts. TSAFE generates two predicted trajectories for each aircraft and checks all four combinations of trajectories for each pair of aircraft. One of the trajectories is synthesized based on the flight plan, tracking data, and atmospheric data. The other trajectory is a “dead-reckoning” trajectory, a short-range projection of position based on the current velocity. Methods were also developed to predict critical leveloff maneuvers and to detect and model unplanned turns. Official reports and tracking data were obtained for 58 actual operational error cases, and an automated system was set up to test TSAFE by replaying the tracking data for each case. The results indicate that TSAFE can provide timely warnings of imminent conflicts more consistently and reliably than Conflict Alert.

INTRODUCTION

The primary job of air traffic controllers is to monitor traffic, detect conflicts, and direct pilots by voice, when necessary, to maintain a minimum separation standard [1]. The minimum separation standard for IFR (Instrument Flight Rules) traffic under the control of Air Route Traffic Control Centers, (ARTCCs, or “Centers”) is 5 nmi horizontally and 1000 ft vertically below FL290 (pressure altitude of 29,000 feet) or 2000 ft at or above FL290 (soon also to be 1000 ft for qualified aircraft). Although mid-air collisions in enroute airspace are extremely rare, incidents in which the minimum required separation is breached are not nearly as rare, and their rate of occurrence has increased significantly in recent years.

When the minimum separation standard is violated

* Aerospace Engineer, Senior Member AIAA, AFC 210-10, Moffett Field, CA 94035, Russ.Paielli@nasa.gov

† Senior Scientist, AIAA Fellow, Heinz.Erzberger@NASA.gov

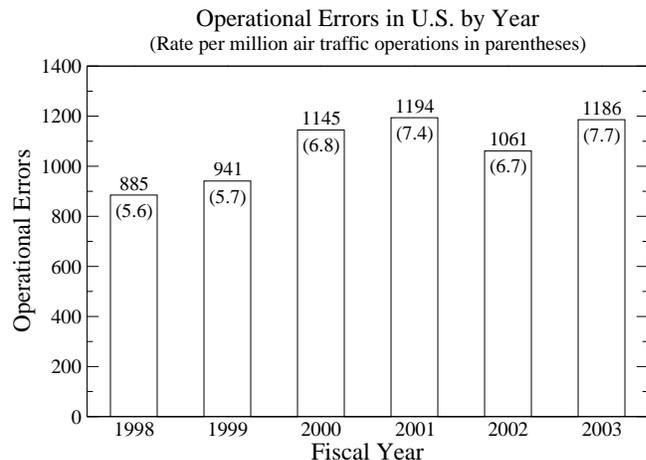


Figure 1: Operational errors in U.S. by year for FY 1998-2003 (ARTCC and TRACON).

in Center airspace, the Center Host Computer System (HCS) detects it using a software “patch” called the Operational Error Detection patch. This software is known more colloquially as the “snitch patch,” and its activation is known on the Center floor as a “deal.” When it happens an investigation ensues (unless it is determined to be a false alarm), and a standardized report documenting and analyzing the incident is written [2]. If a pilot is found to be at fault, the incident is called a “pilot deviation,” but if a controller is at fault it is an “operational error.”

In a recent report [3], the Inspector General of the U.S. Dept. of Transportation wrote, “In terms of safety, FAA and U.S. air carriers have maintained a remarkable safety record. . . . However, operational errors pose a significant safety risk, with an average of three operational errors per day and one serious error (those rated as high risk) every 7 days.” The operational error rate has increased significantly over the past several years, as shown in figure 1 [4].

From FY 1999 to FY 2000, the annual operational error count went from 941 to 1145, an alarming 22% annual increase. In terms of rate per million air traffic operations, that’s a 19% increase from 5.7 to 6.8. (Air traffic operations are counted as one per airplane that

flies in a Center or takes off or lands in a TRACON.) The growth in operational errors continued in FY 2001 even though the last month of the Fiscal Year saw a drastic dropoff in traffic due to the terrorist attacks of September 11th. The rate per million air traffic operations went from 6.8 to 7.4, an increase of approximately 9%. In FY 2002, the error rate per air traffic operation went back down to approximately the FY 2000 level. That could indicate progress or it could simply reflect the fact that error rates (even per air traffic operation) should be less when traffic density is lower. In FY 2003, the error rate per million air traffic operations was back up to 7.7, so the problem certainly remains.

The effort to reduce the rate of operational errors is a multi-faceted endeavor. Selection and training of controllers, disciplinary policies, and facility oversight are certainly important factors, and the FAA is addressing those administrative issues. This NASA study, on the other hand, focuses on the development and testing of a new software tool called TSAFE (Tactical Separation Assisted Flight Environment) to alert controllers of imminent conflicts. TSAFE has been developed as part of the suite of software tools known as the Center/TRACON Automation System (CTAS) [5]. TSAFE is intended to eventually be independent of CTAS, but it is currently one of several possible configurations of CTAS.

TSAFE is intended to enhance or replace a legacy function in the HCS called Conflict Alert (CA), which was installed in the 1970s to warn controllers of imminent potential conflicts. When CA detects a potential conflict, the data blocks of both aircraft blink on the controller's radar display. However, CA sometimes fails to warn of imminent conflicts or warns too late, as will be shown later in the paper. One of the objectives of TSAFE is to improve on the performance of CA by reducing both missed and false alerts and providing earlier alerts when possible.

The official documentation for CA [6] was written long ago and is written from a software perspective rather than an engineering perspective. It is apparently intended for readers who have access to and are familiar with the actual CA software. Also, the HCS software was developed before modern software engineering methods and modern real-time operating systems were available. It is written in an obsolete language (Jovial) and in a non-modular form that makes it difficult to separate from its software environment. More recent documentation on CA [7] does not explain the engineering philosophy or methods used in CA.

Conflict Alert projects the position of each aircraft based on its current velocity, or "dead reckons," for approximately three minutes and checks for predicted loss of separation within that time. It does not project climbing or descending trajectories beyond their cleared

(assigned) altitude, however, because that would cause many false alerts. CA does not account for flight plans in the horizontal plane, however. If a flight plan has a turn at a waypoint, CA simply projects the trajectory straight through the waypoint without turning.

The FAA has provided NASA with access to official investigative reports of operational errors that have occurred in the past four years. These reports were obtained (in the form of paper files) by NASA Ames through its field site at Fort Worth Center (ZFW), which also provided archived tracking and weather data for the operational error cases used in this study. This data was used to test and develop the TSAFE algorithms and software. The results of this analysis are the main subject of this paper.

TSAFE could eventually replace or enhance Conflict Alert. However, TSAFE may be too complex to be incorporated into the legacy software of the Host Computer System, which is based on obsolete software technology of thirty to forty years ago. TSAFE therefore may not be deployable until well after 2010, when ERAM (En Route Automation Modernization) will have replaced the HCS at each Center. However, the FAA recently asked NASA to help them assess whether Conflict Alert can be modified in the HCS to take advantage of some key features of TSAFE before ERAM is deployed. That assessment is currently in progress.

The remainder of the paper is organized as follows. The conflict detection methods used in TSAFE are presented next. Then the operational error analysis and TSAFE replay methods are discussed. Several representative samples of operational error cases are then discussed in detail. Finally, the summary results comparing the overall performance of CA and TSAFE are presented.

CONFLICT DETECTION METHODS

When an operational error occurs, it is usually the result of an air traffic controller failing to detect a conflict with enough lead time to resolve it by maneuvering one of the aircraft before the minimum required separation is breached. Aside from administrative factors involving the selection and training of controllers, the key technical means of reducing operational errors in the near term is to provide timely detection of impending conflicts and appropriate warnings or alerts to controllers.

Conflict detection can be broadly classified as strategic or tactical. Strategic conflict detection is the prediction of potential conflicts between any pair of aircraft for a relatively long period of time, say twenty minutes. Beyond that range, the trajectory uncertainty due to wind modeling error or intent error is usually too large to permit effective conflict detection. Tactical conflict detection, on the other hand, is the prediction of con-

flicts for a relatively short period of time, say three to four minutes.

To be effective, strategic conflict detection must be based on the intended flightpath as specified in the current flightplan, and it must also account for the wind field, which greatly affects the predicted trajectory of the aircraft over the relatively long prediction time. Strategic detection can tolerate a relatively high rate of missed alerts because conflicts can still be caught later by tactical detection. However, a high rate of false alerts would render strategic detection inefficient and disruptive because it would lead to many unnecessary interventions. If an aircraft deviates from its flightplan and its intent is unknown, strategic detection becomes ineffective and is usually abandoned.

Tactical conflict detection, on the other hand, cannot tolerate many missed alerts because, other than the controller, it is the last line of defense against operational errors, with only TCAS (Traffic Alert and Collision Avoidance System) [8] left to prevent collisions. It can tolerate some false alerts, but too many will annoy controllers and desensitize them to valid alerts. Tactical detection can still make use of flightplans and wind data, but its short prediction time makes it less sensitive to that data. More importantly, tactical detection needs to continue even for aircraft that have deviated from their flightplan. Accounting for such deviation is necessarily a heuristic endeavor because it involves guessing where the aircraft might head next. The first guess is always “straight ahead at constant groundspeed,” otherwise known as “dead reckoning.”

A flight might deviate from its flightplan for any of several possible reasons. Inattentive piloting is always a possibility for aircraft that are not using an FMS (Flight Management System). Another common reason is that the flightplan was not updated when a controller issued a clearance. As far as any automated conflict detection is concerned, such a flight is deviating from its intended flightplan even if it is conforming exactly to the voice clearance issued by the controller. In a limited study of one ten-hour period in one particular sector in 1999, Lindsay [9] found that only approximately 30% of (horizontal) route clearances that were issued by voice were actually entered into the HCS (Host Computer System) as flightplan amendments. Conflict detection software would have no indication of the other 70% of those route clearances.

In the same study, Lindsay found that approximately 95% of altitude clearances were entered into the HCS, which far exceeds the 30% for horizontal route clearances. That certainly helps make altitude prediction more accurate and reliable, but problems remain even there. When a controller issues an altitude clearance, the delay before the pilot starts the climb or descent can vary by nearly a minute. In the case of “pilot

discretionary descents” the pilot is given discretion as to when to initiate an arrival descent, and the variation can be several minutes. Obviously, trajectories cannot be predicted with reliable accuracy when such uncertainties exist.

The approach taken in TSAFE to deal with these trajectory uncertainties is to generate two trajectories for each aircraft. One is the strategic trajectory prediction based on the flightplan, altitude clearances, atmospheric data, and radar tracking data. It is computed by a CTAS software process called the Trajectory Synthesizer (TS), so it will be referred to as the TS trajectory. The other is a dead reckoning (DR) trajectory similar to what is used by Conflict Alert, which will be referred to as the DR trajectory. The TS prediction could extend to perhaps twenty minutes, but the focus of this paper is only on the first three minutes. The DR prediction was limited to approximately three minutes, similar to Conflict Alert, to prevent excessive false alerts. In the case of altitude transitions, the DR trajectory also stops at the cleared altitude, as does CA, to reduce false alerts. (The DR trajectory extends past the cleared altitude for critical leveloff prediction, which will be discussed later.) TSAFE checks all four combinations of trajectory types (TS/TS, TS/DR, DR/TS, DR/DR) for conflicts. If one or more of the four combinations show conflicts, an alert is generated.

Figure 2 shows an example of the dual trajectories generated for each aircraft. The top half of the figure shows vertical profiles and the bottom half shows horizontal paths. The TS trajectory is constructed by the Trajectory Synthesizer, and it is based on tracking data (radar and barometric altitude), flightplans, controller clearances (that were entered into the HCS), atmospheric data (winds, pressures, etc.), and aircraft performance models. It can reasonably extend to approximately twenty minutes into the future. Although not shown in the figure, the TS rounds corners to produce flyable turns. The DR trajectory, on the other hand, is based on dead reckoning. It simply projects the current position and altitude based on the current estimated velocity and altitude rate. The projection is limited to approximately three minutes or until the cleared altitude is reached, whichever comes first. The DR trajectory accounts for the possibility that the aircraft may diverge from its flightplan. When an aircraft is flying straight and level in conformance with its flightplan, its DR and TS trajectories will be nearly identical for the first three minutes, but for simplicity both are generated anyway.

The top half of figure 2 shows a climbing aircraft that is cleared to an altitude one flight level below another cruising aircraft. As mentioned above, for purposes of regular conflict prediction, the TSAFE DR trajectory terminates at the cleared altitude, as does the CA trajectory, to prevent excessive false alerting. Actually, the

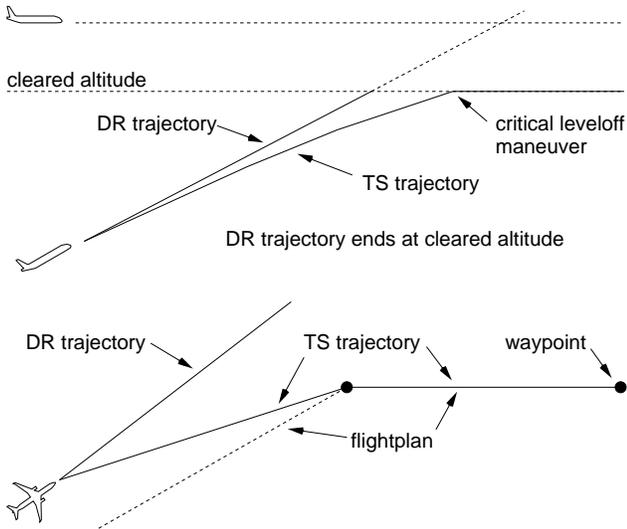


Figure 2: Dual trajectory generation: TS (Trajectory Synthesizer) and DR (dead reckoning). Top: vertical profiles. Bottom: horizontal paths.

TSAFE DR trajectory extends past the cleared altitude, but if a conflict is detected past that point it is classified as a critical leveloff maneuver rather than a conflict. A critical leveloff is a leveloff that must be executed correctly to avoid an immediate loss of separation, as shown in figure 2. [The critical leveloff maneuver is shown on the TS trajectory of figure 2 simply to avoid the need for an extra figure.] Missed critical leveloffs are usually the result of a communication error, examples of which will be discussed later in the paper. Alerting for these cases gives the controller an opportunity to verify that the pilot properly understood the cleared altitude. However, alerts for critical maneuvers need to be distinguishable by the controller from regular conflict alerts.

The question of how to display predicted trajectories and conflicts to controllers is an important one, but it will not be addressed in this paper. If two different trajectory combinations (e.g., TS/TS and TS/DR) result in different conflicts, the conflict that is predicted to occur first would typically be displayed. Also, predicted conflicts could be rated for severity and displayed differently depending on the result, but that will not be addressed here either.

A common cause of operational errors is an altitude clearance that leads directly to a loss of separation. As mentioned above, the delay before the climb or descent is initiated can be uncertain by up to a minute, or several minutes for discretionary descents. Regardless of the delay, or whether or not it is modeled, the TS can generate the climb or descent profile as soon as it is entered into the HCS by the controller. In many cases, nearly instant feedback can be provided on whether the

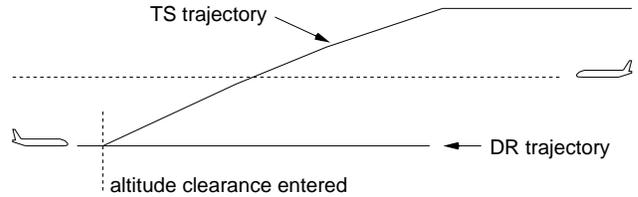


Figure 3: A bad altitude clearance can sometimes be detected with the TS trajectory prediction before the climb or descent is initiated.

clearance is likely to lead to conflict, before the altitude transition is actually initiated, as illustrated in figure 3. If a conflict is detected, the controller can be alerted and can rescind the clearance. This is an example of using the intent information in the strategic TS trajectory for a tactical purpose. It applies to several of the operational error cases examined, and an example will be presented later in the paper.

When an airplane is diverging significantly from its flightplan for some reason, its TS trajectory prediction can be very inaccurate. However, unlike the DR trajectory, the TS trajectory does account for altitude clearances, and these are important even if the aircraft is off its flightplan. As mentioned earlier, altitude clearances are entered by the controller into the HCS much more reliably than horizontal route amendments, so altitude intent is usually known. Hence, a “hybrid” trajectory type is used in place of the TS trajectory when the aircraft is deemed sufficiently out of conformance with its flightplan. The hybrid trajectory uses dead reckoning for the horizontal path, but maintains the altitude profile from the TS.

Another issue that comes up is the noise on the radar tracks and barometric altitude measurements. The noise level combined with the nominal update rate of 12 sec makes altitude rate difficult to estimate accurately, for example. Conventional low-pass dynamic filtering is appropriate most of the time, but in some situations the lag introduced by the filtering is detrimental to rapid detection. In those situations, the filtering was turned off and simple back-differencing of raw altitude measurements was used to determine altitude rate. For example, when an aircraft is approaching a critical leveloff maneuver and a prediction is needed as to whether the aircraft has too much vertical velocity to level off in time, filter lag is unacceptable, so the filter was bypassed. The effectiveness of this and other design features will be shown later in the paper.

Another method that was found effective in TSAFE is the detection and modeling of unplanned turns. After the initiation of an unplanned turn has been detected, the trajectory is turned through a constant radius to “look around the corner.” The challenge is to detect un-

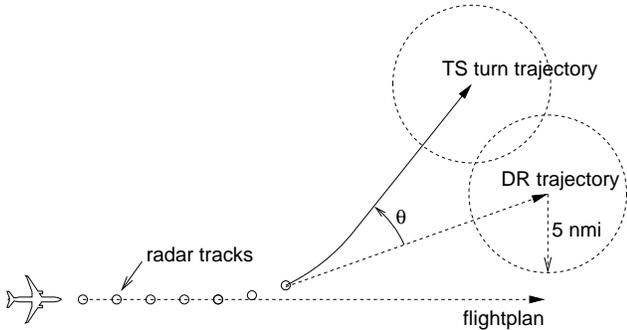


Figure 4: Detection and modeling of unplanned turns.

planned turns as quickly and reliably as possible, given the radar tracking noise and the radar update rate of 12 seconds. Again, conventional low-pass dynamic filtering tends to introduce too much lag. The details of the method used in TSAFE are discussed in the Appendix of reference [10], but a brief outline follows.

The approach that was found to be most effective is illustrated in figure 4. Back-differencing of radar tracking samples is used to estimate turn rate and equivalent bank angle (assuming a coordinated turn). Two consecutive bank angles above a specified threshold are required to establish the initiation of a turn. A horizontal trajectory is then constructed consisting of a turn of angle θ (nominally 35 deg) at constant radius (determined by the tracking data), followed by a straight segment. This turning path is then combined with the TS altitude profile to form a hybrid trajectory as discussed earlier. The turn angle of 35 deg was chosen so that the circles of radius 5 nmi overlap, as shown in figure 4, thereby ensuring that no gap exists between the DR and TS trajectories through which an aircraft could fly without having a conflict detected. An example of an operational error case that benefited from this method will be presented later in the paper.

CATEGORIZATION OF OPERATIONAL ERRORS

As mentioned earlier, official reports of operational errors over the past four years were obtained by NASA through its field site at Fort Worth Center, which also provided archived tracking and weather data. Fifty eight cases have been obtained thus far, with more expected in the future. They were selected strictly on the basis of the availability of reports and data, regardless of the nature of the incidents themselves. These reports are considered sensitive, and care was taken to remove all references to when and where the incidents occurred and who was responsible. The tracking data is in the form of “CMsim”

Operational Errors by Phase of Flight
58 operational error cases

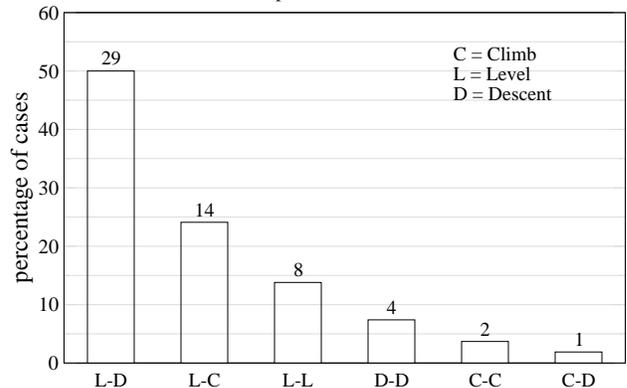


Figure 5: Categorization of operational errors by phase of flight.

files, which are in a format that is used by the CTAS Communication Manager (CM). These files contain the radar tracking data, flightplans, controller inputs, references to weather data files, and other information. They can be “replayed” through TSAFE to determine when and which types of alerts are generated. As mentioned earlier, TSAFE is intended to eventually be independent of CTAS, but it is currently one of several possible configurations of CTAS.

The 58 operational error cases were categorized according to several criteria of interest. Figure 5 shows how they break down by phase of flight. Exactly half (29) of the 58 cases involved a level flying aircraft and a descending aircraft (L-D). The next most common categories were level/climb (L-C) at 24.1% (14) and level/level (L-L) at 13.8% (8). In the remaining 13.0% of the cases, both aircraft were flying nonlevel. Note that the level cases above include aircraft that are flying temporarily level in a longer-term climb or descent. If those are excluded, the number level/level cases drops to 8.6% (5).

The operational error cases can also be categorized as follows:

- 26 (~45%): caused by altitude clearance
- 11 (~19%): critical leveloff maneuver missed
- 6 (~10%): intrail overtake
- 4 (~7%): caused by horizontal route clearance
- 3 (~5%): one or both aircraft in holding pattern

These categories do not cover all cases, nor are they necessarily mutually exclusive. In many cases, the controller actually *caused* the loss of separation by issuing a bad clearance. If such a clearance was issued less than three minutes before loss of separation, it was counted as the

cause, otherwise it was not. Altitude clearances were the most common type of clearance that caused loss of separation. That is not surprising because altitude clearances are very common, and controllers have no graphical representation of altitude in their planview display (they rely on the altitude field in each aircraft data block). Altitude clearances were found to cause 26 (44.8%) of the cases. In 11 cases (19.0%) a critical leveloff maneuver was missed, 7 due to a misunderstood altitude at the arrival meter fix, 3 due to a misstated altitude clearance by the controller, and 1 due to a misunderstood aircraft call sign. Six cases of intrail overtakes occurred, four merging arrival overtakes, and two climbing intrail overtakes. In four cases, bad horizontal clearances were issued for various reasons, such as weather avoidance and metering delay. Three cases involved holding patterns, two of which had both aircraft flying level.

REPLAY METHODS

The CMSim file for each operational error case was “replayed” in the TSAFE configuration of CTAS to determine when and which types of alerts are generated. The replaying of 58 operational error cases could be very tedious, especially considering that it must be done over again each time a significant revision is made to the TSAFE software being developed and tested (typically about once per week). In the TSAFE configuration, CTAS involves five separate software processes. Starting them all manually for each case would take exorbitant amounts of someone’s time, so a recursive shell script was written in BASH (a standard shell scripting language) to automate the startup procedure. But just executing the startup script manually for each of 58 cases would still consume excessive amounts of time. Hence, an executive BASH script was developed to invoke the CTAS startup script for each case, let it run to completion, shut down CTAS, archive the output data, then start CTAS again for the next case. This automated batch processing, which had never before been done with CTAS, was essential to the development and testing of TSAFE.

The recorded CMSim files for each case contain data for all the traffic that was running at the time, which could be several hundred aircraft. To greatly reduce the computational load of the replays, a script was written in Python (a free scripting language) to extract data for only the two aircraft involved in the operational error. Several other Python scripts were written to plot the actual trajectories of the two aircraft, the predicted trajectories from TSAFE, and a record of the resulting alerts from TSAFE (the actual plots were generated using a free, open-source plotting program called GRACE). The plotting was automated as part of the batch processing, so that the plots for all 58 cases were automatically regenerated for each batch run. The only part of the

plotting that was not automated was transcription of controller voice commands and other events from the narrative in the official report of each case.

In the course of this automation effort, several logistical challenges arose. The CMSim file formats changed over the years, for example, and the Python scripts had to be able to handle all the variations. Another problem was caused by the Chart Change Updates (CCU) that the FAA applies on a cycle of 56 days. The CCUs involve changes in various geographic parameters, such as waypoint locations and the reference tangent point of the stereographic (pseudo-Cartesian) coordinate system for each Center. The problem is that the tracking data in the CMSim files is in stereographic coordinates (x,y), but the flightplans contain waypoints that are specified in terms of geodetic coordinates (latitude and longitude), which get converted to stereographic at run time. If the current reference tangent point differs from the one that was valid at the time of the incident (almost always), the resulting flightplans could be offset by several nautical miles, which would cause errors in the predicted TS trajectories. The solution was to use the CTAS software version control system to automate the retrieval of archived map parameters and other site “adaptation” data that was valid when the incident occurred.

The horizontal separation criterion used for all results presented in this paper was the standard legal minimum of 5 nmi unless stated otherwise. This value would probably be increased slightly (perhaps to 5.5 or 6 nmi) in practice to further reduce missed alerts. However, it was left at 5 nmi here to provide a fair comparison with CA. The vertical separation criterion was the standard 1000 ft (or 2000 ft above FL290) if both aircraft were flying level. However, if one or both aircraft were in altitude transition, the vertical criterion was expanded slightly to 1200 ft (or 2200 ft above FL290) to account for the additional altitude uncertainty during transition.

SAMPLE OPERATIONAL ERRORS

Of the 58 operational error cases that were examined, several were selected as representative of the most common types. A few key plots of the representative cases will be presented. As mentioned, all plots were automatically generated except for the transcription of discrete events (e.g., voice clearances) from the official reports. However, the plots to be presented in this paper were modified slightly from the automated plots to remove aircraft call signs and other potentially sensitive information.

Sample Case 1: Altitude Clearance

In the first sample case, a sequence of two route clearances and an altitude clearance combined to cause the

loss of separation. The altitude clearance was the final cause. The top plot of figure 6 shows the groundtracks of the two aircraft involved. Aircraft 1 (AC1), represented by the dark solid line, is heading southwest, and aircraft 2 (AC2), represented by the dark dashed line, is heading northeast. The flightplans for each aircraft are shown as gray lines of the same type as the corresponding aircraft tracks (solid or dashed), with a gray “×” marking each waypoint. The two overlapping circles with asterisks inside them are five nmi in diameter (or 2.5 nmi radius, so they become tangent when horizontal separation is 5 nmi) and indicate the aircraft positions at the first radar update following loss of separation.

In this case, four time-tagged events are each marked by a “+” on the groundtrack plot. The times are relative to loss of separation. At -4:43, AC1 requested a vector for arrival, and 40 seconds later at -4:03 was told to turn 20 deg right. This vector clearance, which was not entered into the HCS (Host Computer System), led AC1 toward the path of AC2. Thirty six seconds later, at -3:27, the controller detected the potential conflict and vectored AC2 20 deg right. This would have resolved the conflict, but the controller prematurely cleared AC2 back to a fix at -1:47, turning it back toward AC1. The two aircraft had been flying level at different altitudes, but at -2:04 AC1 was cleared to descend to FL280, sending it through the cruising altitude of AC2.

The altitude profiles are shown in the middle plot of figure 6. AC2 was flying nominally level with some unsteadiness in altitude, and AC1 was flying level until approximately two minutes before loss of separation. At -2:04, AC1 was given a temporary altitude clearance to descend to FL280, which the controller entered into the HCS. The gray line shows that the cleared altitude went down to FL280. This descent clearance, in combination with the direct-to-fix clearance mentioned above, caused the operational error. Had it not been issued, the vertical separation would have stayed at 2000 ft as the two airplanes passed by each other.

The bottom plot of figure 6 shows the two-dimensional separation near the encounter for sample case 1, where each radar hit is represented by a square. The origin represents a collision. In this case the two airplanes came within approximately 3.4 nmi horizontally and 300 ft vertically of each other. Figure 7 shows the alerts generated by TSAFE. It also shows the time at which Conflict Alert activated, which was taken from the official report of the incident. The time axis is relative to loss of separation. The dashed vertical line indicates that CA did not activate until approximately -0:01, or one second before actual loss of separation. This CA alert is obviously too late to prevent a loss of separation (but could still help prevent a collision). As for TSAFE, it alerted with codes 2 (TS/DR) and 4 (DR/DR) at approximately one minute before loss of separation, which

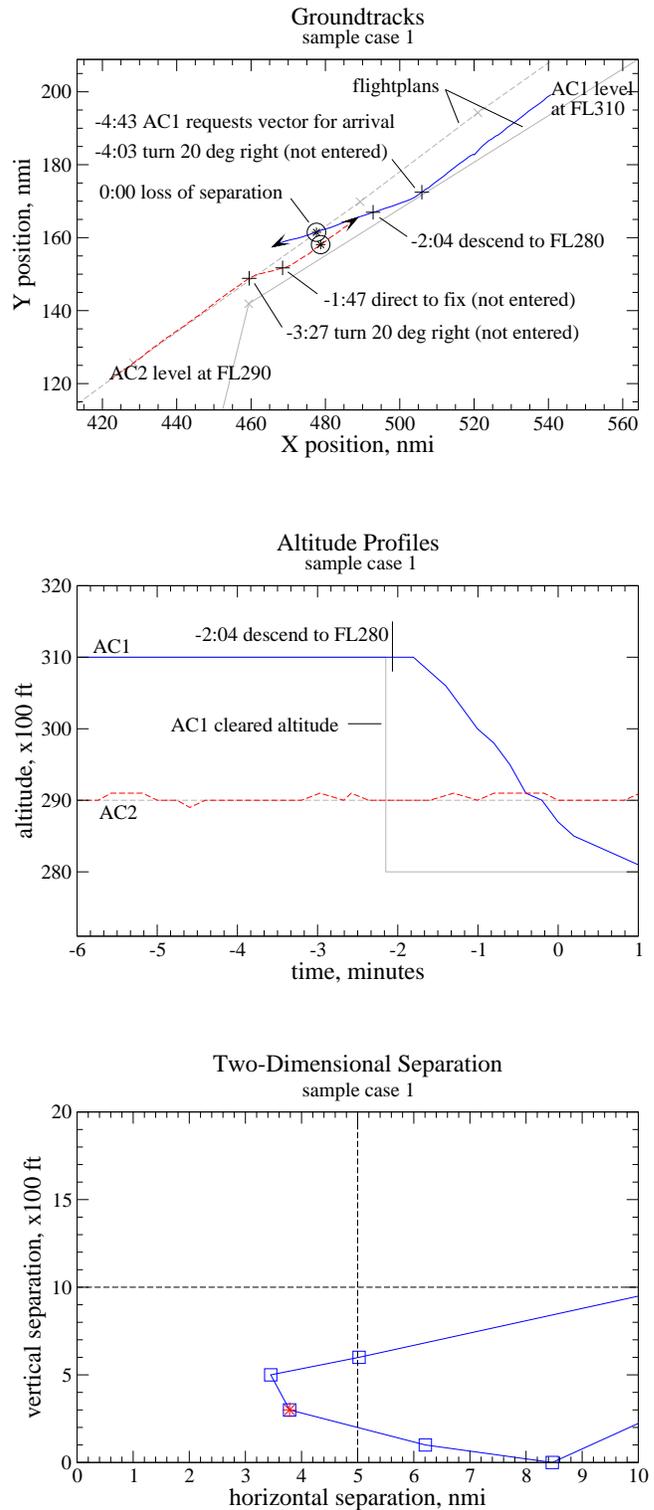


Figure 6: Groundtracks, altitude profiles, and two-dimensional separation for sample case 1.

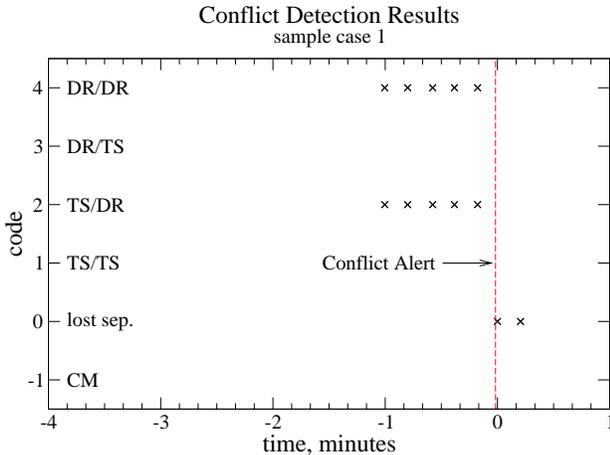


Figure 7: Conflict detection results for sample case 1.

would have likely been early enough to prevent the loss of separation.

Sample case 1 is an example of the TSAFE dead reckoning outperforming CA. Since CA is also based on dead reckoning, the reason that TSAFE provided a much earlier alert is not obvious. A possible explanation is that the velocity estimation algorithms [11] used in CTAS (and TSAFE) perform better than the corresponding algorithms used for the dead reckoning in CA. However, we do not currently have access to the data needed to test this hypothesis.

Sample Case 2: Altitude Clearance

Like sample case 1, sample case 2 is an operational error that was caused by an altitude clearance. Whereas the conflict in sample case 1 was detected by TSAFE based on the DR trajectories, the conflict in sample case 2 was detected based on the TS trajectories. The TSAFE alerts occurred shortly after the clearance was entered into the HCS but *before* the aircraft actually began the altitude transition (descent in this case). This phenomenon occurred in 15 of the 26 cases caused by altitude clearances, or approximately 26% of the entire set of 58 cases.

The top plot of figure 8 shows the groundtracks. The triangle indicates the point at which control was handed to ZFW (slightly before the Center boundary, as is typical). AC1 was following its flightplan very closely, and AC2 was also following its flightplan but is off by a couple of nautical miles. Again, AC1 and AC2 started out flying level at different altitudes, but at -1:51 AC1 was cleared to descend to FL240, which sent it through the cruising altitude of AC2. At approximately one minute before loss of separation, AC1 was vectored to the right to avoid AC2, but the maneuver was too late to avoid loss of separation. (This is one of the few cases

for which the official report is not yet available, so the controller voice clearances are not known exactly.) The minimum separation for this case was approximately 3 nmi horizontally and 800 ft vertically.

The altitude profiles are shown in the bottom plot of figure 8. AC2 was flying level at FL330 throughout, and AC1 was flying level at FL350 until -1:51, at which time it was issued a temporary altitude clearance to FL240. The vertical gray line indicates that the controller entered this clearance into the HCS (it goes down to FL240, which is off the plot). The resulting predicted altitude profiles for AC1 are shown in figure 9. The circles in this figure each represent a barometric altitude sample. The solid lines emanating from each circle are the predicted altitude profiles. Again, the vertical gray line indicates the temporary altitude clearance, which was entered at approximately -1:49. At that time, the aircraft was predicted to start descending immediately to FL240. The predicted descent profiles were considerably steeper than the actual descent due to modeling error.

The six-pointed (red) stars indicate points of predicted loss of separation, which were all approximately 20 s before actual loss of separation due to trajectory prediction error. Actually, each star is an upward triangle and a downward triangle at the same point. The former indicates predicted loss of separation with the DR trajectory of the other aircraft, and the latter indicates the same for the TS trajectory of the other aircraft (because AC2 is flying straight and level, its TS and DR trajectories are virtually identical). The key point here is that TSAFE provided an alert almost instantly after the altitude clearance was entered at approximately -1:49. On the other hand, CA did not activate until +0:08, or 8 sec after loss of separation. The intent information in the TS trajectory was critical to the timely detection of this conflict.

Sample Cases 3 and 4: Critical Leveloff

Sample cases 3 and 4 are representative of the class of operational errors involving a missed critical leveloff maneuver. A critical maneuver is a maneuver, such as a turn or leveloff at a specified altitude, that must be executed as planned to avoid an imminent loss of separation. Critical leveloff maneuvers that were missed account for 11 of the 58 operational error cases, or approximately 19%.

The groundtracks are not shown for sample case 3. The two aircraft were converging at an angle of approximately 45 deg. The altitude profiles are shown in the top plot of figure 10. AC2 was flying level at FL250, and AC1 was climbing to FL280. At -2:49, the controller cleared AC1 to climb to FL250 even though he had entered FL240 into the HCS, as indicated by the gray line (apparently the controller entered the value about a minute and a half before issuing the voice clearance). At

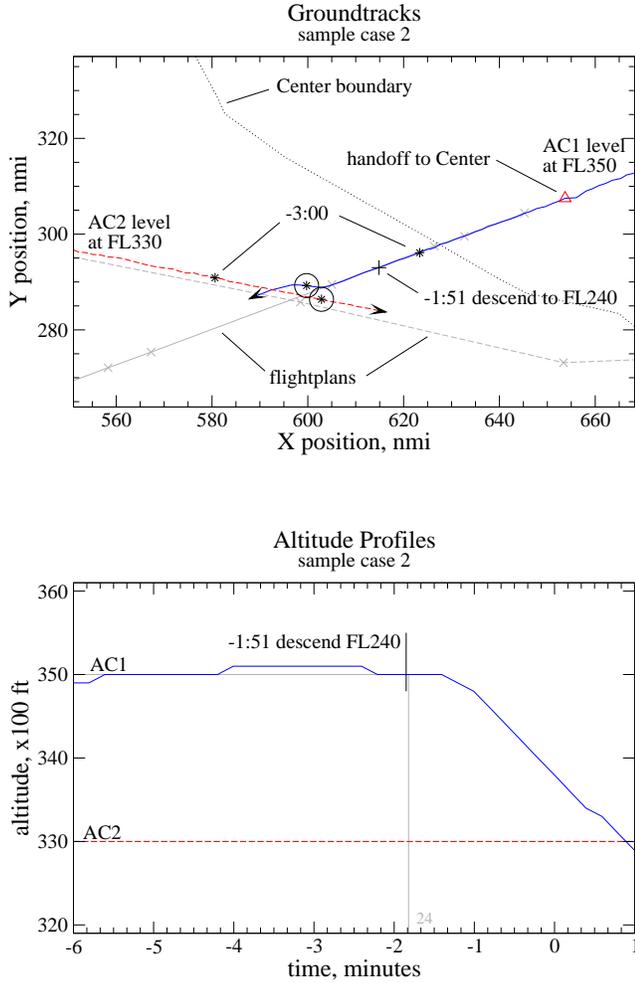


Figure 8: Groundtracks and altitude profiles for sample case 2.

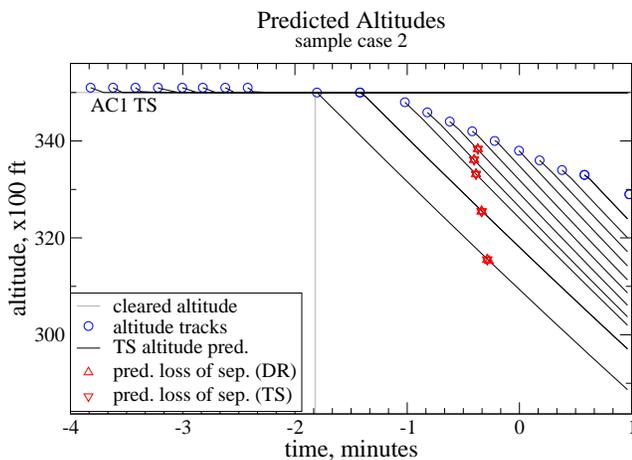


Figure 9: TS predicted altitudes for sample case 2.

0:00, the point of loss of separation, the controller recognized the problem and asked the pilot to “verify you’re level at FL240,” which indicates that the controller really meant to clear AC1 to FL240 but had erroneously said FL250.

The critical maneuver in this case is the leveloff at FL240, which needed to be executed to avoid an immediate loss of separation. The problem was that the pilot thought he was cleared to FL250. Had the critical maneuver been indicated on the display, the error could have been corrected in time to avoid the loss of separation. Had the controller been alerted in advance to the impending critical maneuver, he could have confirmed the cleared altitude, which would have uncovered the misunderstanding earlier. The bottom plot of figure 10 shows the TSAFE alerting results. Code -1 indicates that a critical maneuver has been detected. As soon as the temporary altitude of FL240 was entered at approximately -4:22, the critical maneuver was detected.

Note also that, for this particular case, code 3 (DR/TS) and 4 (DR/DR) alerts were generated at approximately -0:24, whereas CA did not activate until -0:01. Thus, TSAFE provides 23 sec of additional lead time, which in this case could have been critical. The superior performance of TSAFE is due, in part, to a simple algorithm that turns off the altitude rate filtering (to eliminate the filtering lag) and estimates the rate of climb (or descent) just prior to the cleared altitude, as discussed earlier. If that altitude rate is so high that the aircraft cannot possibly level off at the cleared altitude, then an alert is generated. This logic provides additional lead time compared to simply waiting for the cleared altitude to be penetrated. Note, however, that the potential lead time is limited by the Mode-C altitude update rate (once per 12 sec) and the substantial noise on the baro-altimeter measurements.

Several other critical maneuver cases involve descents to an arrival meter fix. In some cases, the crossing altitude at the meter fix was misunderstood, and in other cases the aircraft call sign was misunderstood. Sample case 4 is a typical example of a misunderstood crossing altitude at an arrival meter fix. The groundtrack plot is not shown here, but the two airplanes were merged for arrival when the loss of separation occurred. The altitude profiles are shown in figure 11. At -5:31, the controller told the pilot to cross the meter fix at 11,000 ft, but the pilot read back 10,000 ft. The controller did not catch the error. The misunderstanding did not become apparent until separation was already lost. The gray line shows the meter fix altitude that was entered at 11,000 ft. (The downward triangles indicate handoff to the TRACON, but the error was still charged to the Center controller.) Although the relative velocity was fairly small, the minimum separation got down to slightly over 1.0 nmi at the same altitude, which is con-

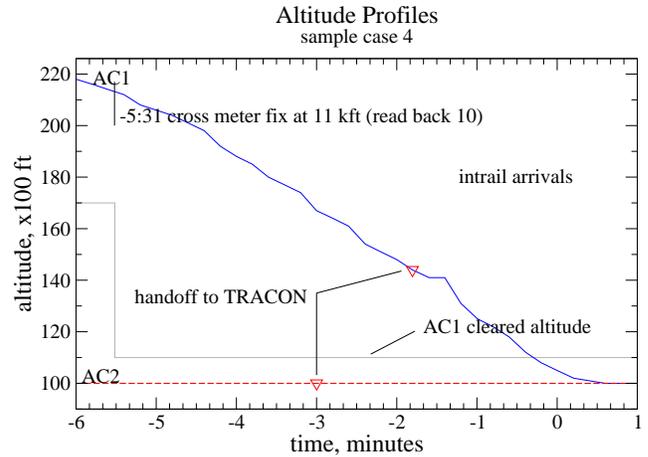
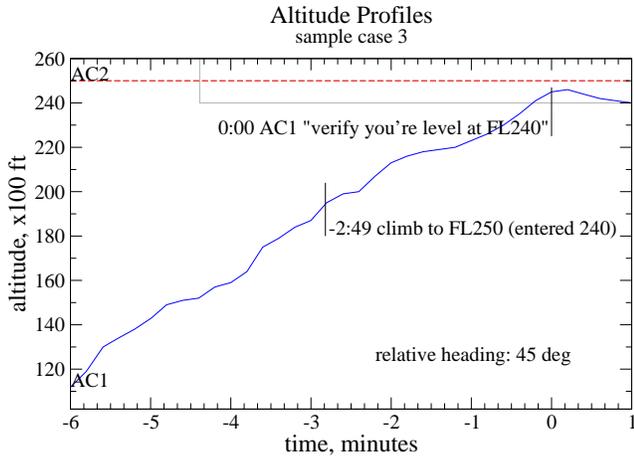


Figure 11: Altitude profiles for sample case 4.

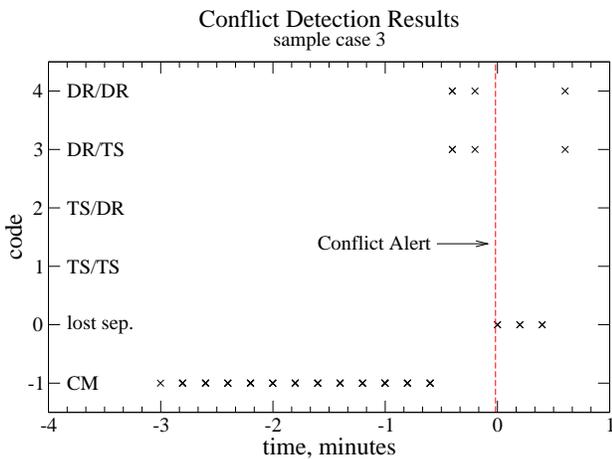


Figure 10: Altitude profile (top) and conflict detection results (bottom) for sample case 3.

sidered severe. Had TSAFE been in use, the critical maneuver prediction feature could have prompted the controller to confirm the crossing altitude, and the error might have been prevented. CA did not activate until separation was lost.

Critical maneuvers are not actual predicted conflicts but potential conflicts that will occur only if the critical maneuver is not executed properly. If treated as ordinary predicted conflicts, nearly all of them will be false alerts. Hence critical maneuvers should be indicated on the display in such a way as to distinguish them from ordinary conflicts. They should also perhaps be indicated only when the predicted horizontal separation is less than, say, 2 or 3 nmi. That is, they should perhaps be limited to severe cases in which the probability of collision is significant. Furthermore, critical maneuver alerts should probably be limited to 2 minutes or less before the critical maneuver is expected to occur.

Sample Case 5: Unplanned Turn

Sample case 5 is representative of cases in which an unplanned turn occurs. The top plot of figure 12 shows the groundtracks. Both aircraft were in holding patterns in level flight at the same altitude (FL310). The flightplans are not shown because they do not apply during holding patterns. The middle plot of figure 12 shows the predicted ground tracks for AC2. The circles represent the radar tracks, which also mark the beginning of each predicted trajectory. The solid lines that turn to the right (the aircraft's left) are the predicted TS trajectories based on the flightplan, which was still active in the HCS because the controller hadn't entered the holding pattern. The dashed lines indicate that either (1) the aircraft deviated from its flightplan or (2) an unplanned turn was detected. In the former case, a horizontal dead-reckoning trajectory is used, and in the latter case a turn of 35 deg is generated based on the observed turn direction and rate. In either case, the resulting horizontal path is combined with the TS altitude profile to form a hybrid trajectory.

Figure 12 shows two spuriously detected left turns followed by several straight DR trajectories, then the correctly detected right turn. (The first spuriously detected left turn could be the start of an aborted turn to the next waypoint.) The detected right turn was modeled with an additional 35 deg of predicted heading change as discussed earlier. This turn look-ahead capability provided additional lead time in predicting the conflict. The downward triangles indicate a predicted loss of separation with the TS trajectory of AC1, and the upward triangles indicate the same with the DR trajectory of AC1. The conflict detection results are shown in the bottom plot of figure 12. The points at code 1 show that the turn detection and modeling method yielded an additional 36 seconds (3 radar samples) of alert lead

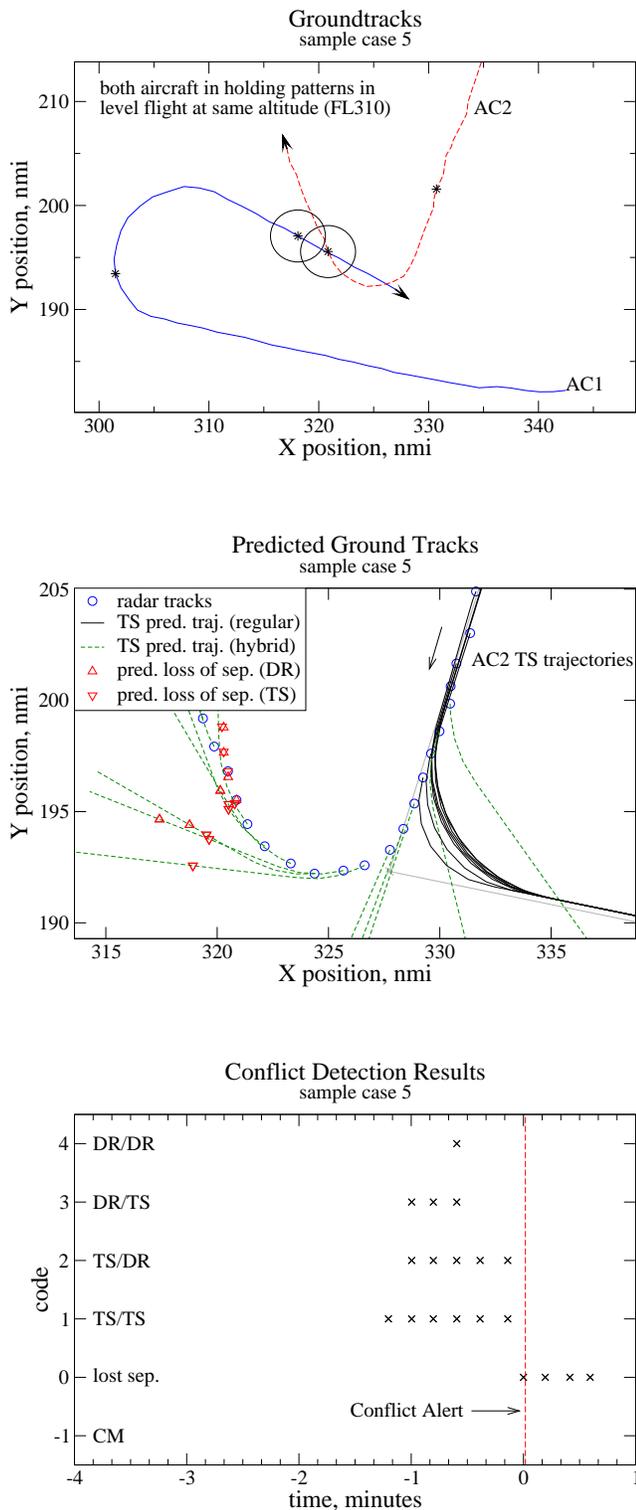


Figure 12: Groundtracks (top), predicted ground tracks (middle), and conflict detection results (bottom) for sample case 5.

time compared to the code 4 alerts based on dead reckoning. CA did not provide an alert until slightly after separation was actually lost.

SUMMARY RESULTS

In the preceding section, the alerting performance of CA and TSAFE was compared for several sample cases of operational errors. But how do CA and TSAFE compare for all 58 cases? A couple of plots will be used to answer that question. The plots are based on the alerting lead times for CA and TSAFE. The lead time for CA was taken directly from the official FAA report for each case. For TSAFE, the lead time is based on the first alert within three minutes of the actual loss of separation, regardless of type (TS/TS, TS/DR, DR/TS, or DR/DR). Critical maneuver alerts are counted separately from ordinary conflict alerts. If the alerts are interrupted by a time gap that exceeds a parameter called the gap tolerance, then the alerts before the gap are not counted. The time gap tolerance used for this paper was two minutes.

Figure 13 shows a scatter plot of the alert lead times for CA vs. TSAFE. It shows a discrete “x” symbol for each case in which both TSAFE and CA provided an alert. The horizontal position on the plot represents the CA alert lead time, and the vertical position represents the TSAFE alert lead time. The gray 45-deg line is the line of equal performance. Note that the axes go negative because in several cases CA did not activate until *after* loss of separation (a negative alert lead time). In several cases CA did not activate at all (according to the official reports), and those cases are represented by a circle placed at an arbitrary CA lead time of -20 sec (20 sec after loss of separation) so they appear on the plot. TSAFE provided an earlier alert than CA in all but 2 of the 58 cases, though 2 more were nearly equal. For the two cases in which CA provided an earlier alert than TSAFE, TSAFE still provided approximately two minutes or more of lead time, which is adequate.

The other summary results are the cumulative alert lead time data shown in figure 14 and table 1. The cumulative alert percentages of figure 14 are the percentage of operational error cases for which a candidate conflict detection method provides an alert lead time greater than or equal to the ordinate value. Table 1 gives a few discrete numerical points from figure 14. Table 1 shows, for example, that CA provided a lead time of 1.0 minute or more for 12.1% of the 58 cases, whereas TSAFE did so for 70.7% of the cases. When critical maneuvers are added in (TSAFE+CM), performance increases to 82.8% of the cases for 1.0 minute or more of lead time.

In figure 14 and table 1, TSAFE/DR means that only alerts based on pure dead-reckoning (DR/DR) trajectories are counted (not TS trajectories). Since CA is also based on dead reckoning for approximately three

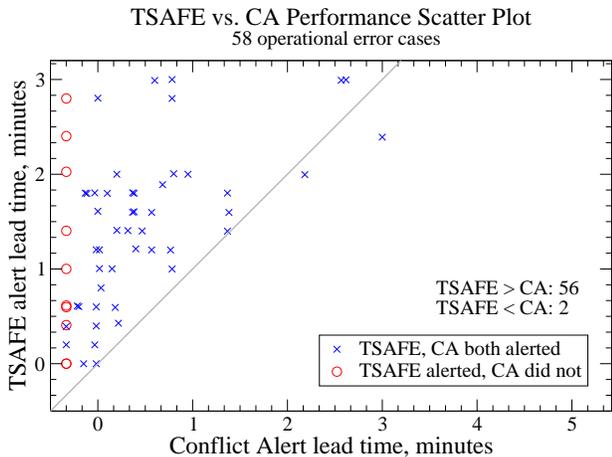


Figure 13: TSAFE vs. CA alert lead time performance scatter plot.

	lead time, minutes			
	>0.0	>0.5	>1.0	>2.0
CA	60.3	29.3	12.1	6.9
TSAFE	100.0	82.8	70.7	22.4
TSAFE+CM	100.0	93.1	82.8	44.8
TSAFE/DR only	77.6	58.6	41.4	12.1

Table 1: Cumulative alert lead times, percent of 58 operational error cases (CM=critical maneuver).

minutes, the TSAFE/DR performance should be approximately the same, but it is clearly better. The reason for this difference is not known for certain at this time, but as mentioned earlier, it could be due to better velocity (groundspeed, course, and altitude rate) estimation in CTAS (and TSAFE) compared to the HCS (and CA). However, we do not currently have access to the data needed to test this hypothesis.

The results show that TSAFE provided an alert lead time of zero minutes or more for all cases, but CA did so for only 60.3% of cases. An alert lead time of zero is obviously insufficient to avoid a loss of separation, but it could be critical to preventing a *collision*. Hence, the fact that CA fails to provide an alert before loss of separation in nearly 40% of cases is significant and somewhat puzzling. Figure 14 shows that the curve for CA is steep in the region near zero alert time (loss of separation). Within a few seconds after loss of separation the alert rate is up to approximately 70%. Within another 15 seconds it is up to approximately 80%, but that still leaves approximately 20% of cases without alerts 15 seconds after loss of separation (note that the loss of separation may have lasted less than 15 seconds in some cases). As mentioned earlier, the CA activation times are all based on the official reports for each operational error case.

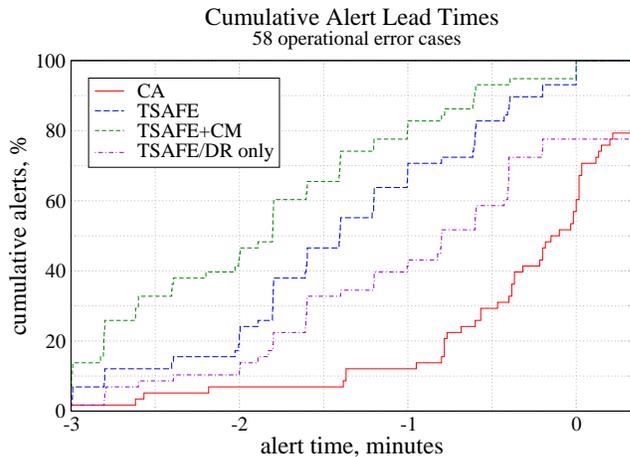


Figure 14: Cumulative alert lead times: percentage of cases with specified alert lead time *or better* (CM=critical maneuver).

CONCLUSION

A new tactical conflict detection tool called TSAFE (Tactical Separation Assisted Flight Environment) has been developed and tested. It generates two predicted trajectories for each aircraft, one based on flightplan intent and the other based on dead-reckoning, and it checks all four combinations of trajectories for each pair of aircraft. It was tested by replaying tracking data from actual operational errors and comparing the alert lead-time performance with that of Conflict Alert. The results indicate that TSAFE can provide timely warnings of imminent conflicts more consistently and reliably than Conflict Alert.

The performance of any detection scheme depends on the rate of false alerts as well as missed alerts. In this study, various methods were used to keep false alerts to a minimum. For example, the nominal separation criteria were kept at the legal minimum of 5 nmi horizontally and near the legal minimum of 1000 or 2000 ft vertically. Also, the dead-reckoning trajectories were limited to three minutes and were not allowed to pass the cleared altitude during altitude transition. They could probably be reduced to two minutes to further reduce false alerts without substantially degrading performance, if necessary. False alerts rates were not addressed directly in this paper, but a study is ongoing to quantify them, and preliminary results are encouraging.

References

- [1] Air Traffic Control FAA Order 7110.65, *Federal Aviation Administration*, Washington, DC.
- [2] Air Traffic Quality Assurance FAA Order 7210.56, *Federal Aviation Administration*, Washington, DC.
- [3] “Key Issues for the Federal Aviation Administration’s FY 2005 Budget,” Kenneth M. Mead, Inspector General, U.S. Dept. of Transportation, April 22 2004.
- [4] “Operational Errors and Runway Incursions: Progress Made, But the Number of Incidents is still High and Presents Serious Safety Risks,” Federal Aviation Administration Report Number: AV-2003-040, April 3, 2003
- [5] Denery, D. G. and Erzberger, H.: “The Center-TRACON Automation System: Simulation and Field Testing,” Proceedings of the advanced Workshop on ATM (ATM 95) sponsored by the National Research Council of Italy; Oct. 2-6, 1995; Capri, Italy; Also published as NASA Technical Memorandum 110366, August, 1995. See also <http://www.ctas.arc.nasa.gov/>
- [6] National Airspace System En Route Configuration Management Document, Host – A5f1.3 – Computer Program Functional Specifications, NAS-MD-321 Automatic Tracking, *Federal Aviation Administration*, 2002.
- [7] *Functional Audit of Existing En Route Capabilities for the En Route Software Development and Support III (ERSDS III) Contract*, prepared for the Federal Aviation Administration by Computer Sciences Corp., July 2001, Section 5.5.4 and Appendix J.
- [8] Introduction to TCAS II Version 7. Federal Aviation Administration, Nov 2000.
- [9] Lindsay, K.S.: “Currency of Flight Intent Information and Impact on Trajectory Accuracy,” presented at FAA/Eurocontrol Technical Interchange Meeting on Shared Flight Intent Information and Aircraft Intent Data, Atlantic City, NJ, Oct 2000.
- [10] Erzberger, H.: “Transforming the NAS: The Next Generation Air Traffic Control System,” NASA/TP—2004-212828, in press.
- [11] Goka, T.; Robinson, J.E.; Eshow, M.M.: “Course Estimation and Turn Detection algorithms for the Center-TRACON Automation System,” NASA/TM—2004-212835, in press.

last revision: 2004-09-09