DESIGN OF A CONFLICT DETECTION ALGORITHM FOR THE CENTER/TRACON AUTOMATION SYSTEM

Douglas R. Isaacson and Heinz Erzberger

NASA Ames Research Center
M/S 210-9
Moffett Field, CA 94035-1000
USA

ABSTRACT

An algorithm for detecting and analyzing potential enroute conflicts has been designed and implemented within the Center-TRACON Automation System (CTAS). The design uses the 4D trajectories provided by CTAS to produce a set of probable future conflicts, and assists the enroute sector controller in efficiently resolving these conflicts. Conflicts are detected via comparisons of trajectories at closely spaced time instants, with measures taken to limit the computations required to complete the search. Performance tests indicate more than 800 aircraft can be processed by the conflict detection and analysis algorithm within a search cycle of 10 seconds. This suggests that the search algorithm easily meets the performance requirements for an automated conflict detection and resolution tool in the current air traffic system. The algorithm includes a trial resolution functionality which automatically detects conflicts of proposed resolutions, and gives near instantaneous feedback to controller input. Field evaluation of the Conflict Probe will be conducted at the Denver and Fort Worth Air Route Traffic Control Centers (ARTCC) beginning in September, 1997.

INTRODUCTION

The current U.S. air transportation system relies on a fixed network of jet airways to maintain a safe and orderly flow of traffic. This network imposes restrictions which lead to less efficient routing than direct or wind-optimal trajectories. Advances in navigation, communication, and automation technology will allow relaxation of these routing restrictions, and thus increase efficiency, without compromising safety. Ultimately, it is envisioned that operators will have the authority to select aircraft trajectories in real time, thus eliminating the need for a fixed network of jet routes everywhere except in terminal areas surrounding large airports. This concept is referred to as "Free Flight" [1,2].

In the absence of a fixed network of routes, controllers will need automated conflict detection and resolution tools, collectively referred to as a Conflict Probe, to maintain situational awareness. Conflict detection consists of three steps: trajectory prediction, conflict pair identification, and conflict probability analysis [3]. The problem of accurate trajectory prediction for departure, en-route and arrival traffic has been solved by the NASA/FAA Center/TRACON Automation System (CTAS) [4,5]. Conflict detection requires that all aircraft trajectories be compared against one another for potential conflicts. A conflict is defined as a loss of required minimum separation between two or more aircraft. The minimum allowable horizontal separation for enroute aircraft is currently 5 nautical miles (nmi); the required vertical separation above flight level 290 (29,000 feet (ft)) is currently 2000 ft, and 1000 ft. below that level. The purpose of conflict probability analysis is to evaluate each potential conflict through the use of conflict probability estimates [3,6]. This paper focuses on the design of the conflict detection algorithm which employs the accurate 4D trajectories provided by CTAS.

The paper begins by defining the requirements of a conflict probe and discussing the impact of these requirements on the design and implementation of the conflict detection algorithm. The key features of the detection process are discussed, including the 4D trajectory search algorithm. Integration of conflict probability estimates into a probability filter are briefly discussed, followed by a description of a conflict resolution functionality referred to as trial planning. Finally, some observed performance results are presented of the conflict probe operating with live traffic from the Denver Center.

CONFLICT DETECTION REQUIREMENTS

The purpose of a conflict probe is to assist the controller in resolving future conflicts efficiently; the purpose of the underlying conflict detection algorithm is to identify all future conflict pairs. This is accomplished by comparing predicted trajectories for all aircraft in the given airspace. To provide controllers with the necessary assistance in a timely and effective manner, four requirements are provided: efficiency, flexibility, completeness, and trial planning capability.

Efficiency is the most influential factor in the design of the conflict prediction algorithm. The computational magnitude of this problem can be seen by noting that the maximum number of pair-wise trajectory comparisons required for n aircraft is equal to the number of distinct pairs of aircraft in a group of n and is found by combinatorial analysis to be $n(n - 1)/2$. This quadratic growth of trajectory pairs with number of aircraft to be searched requires careful algorithmic design that maximizes computational speed. Furthermore, enough time must remain in the radar update cycle for inter-process communications, advisory processing, and trajectory updates. Future conflict probe functionality will also include automated resolution advisories, which further limit the time allowed for conflict detection.

The conflict probe must accommodate varied controller preferences and operational procedures encountered at different enroute facilities. Furthermore, the underlying detection and analysis algorithm must be extendible to meet the requirements of future applications such as automated resolution advisories and, eventually, airborne based detection and resolution systems. Therefore, key conflict search parameters, such as separation requirements and detection time horizons, should be user-definable. Flexibility is also enhanced by the ability to repeat the conflict search whenever radar updates of track positions are received. This ensures that new conflicts are detected and resolved conflicts are discarded at the earliest possible time. Thus, to ensure maximum flexibility, the conflict search cycle should be repeated for every aircraft in less than the radar update cycle of approximately 12 seconds.

Acceptability of any automated conflict detection tool is ultimately determined by the controllers who use it. This decision will be based largely on the completeness and dependability of the system in the operational environment. Two key factors determining acceptability are directly related to the conflict detection and analysis algorithm: missed alert rate and false alert rate. Both missed alerts and false alerts are consequences of the uncertainty involved in conflict prediction.

A missed alert occurs when evasive action is required by the controller or pilot to avoid a conflict that was not detected by the conflict probe. In the absence of a conflict probe, controllers react to potential conflicts only a few minutes before they would occur. Because controllers are ultimately responsible for maintaining aircraft separation, any conflict detected by an automated tool, displayed to the controller before it would otherwise be recognized, is an improvement over the current system. It should be noted that a missed alert in itself does not lead to an operational error (loss of legal separation). An operational error represents a failure not only in automated conflict detection, but also in the three currently existing means of preventing conflicts: manual conflict detection by the controller, near term conflict alert, and the Traffic Alert and Collision Avoidance System (TCAS). Therefore, missed alerts, while undesirable, are of less importance to the design of the Conflict Probe than false alerts, as long as missed alerts occur well before the controller would normally detect the potential conflicts by visually scanning the radar display.

False alerts are predicted conflicts that, in the absence of controller or pilot actions, would not actually materialize. False alerts lead to unwarranted evasive maneuvers, thus wasting fuel and unnecessarily increasing controller workload. Thus, minimizing the false alert rate is an essential design requirement for a conflict detection algorithm.

Controllers resolve conflicts by issuing clearances (speed, altitude, or route changes) to one or more aircraft involved in the conflict. It is difficult for controllers to devise an efficient resolution for conflicts more than a few minutes into the future. Trial planning is the process of proposing a resolution, automatically checking the proposal for conflicts, and allowing the controller to modify the proposal until a conflict free resolution is achieved. Trial planning should be accomplished quickly, and with little workload.

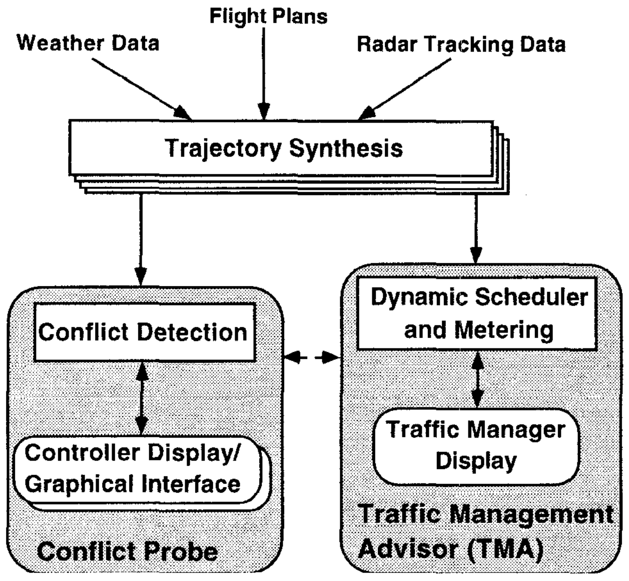


Figure 1. CTAS-Conflict Probe Architecture

CONFLICT DETECTION ALGORITHM

The conflict search algorithm is a fundamental component of the Conflict Probe, producing a set of probable future conflicts displayed to the controller. The foundation of all CTAS processes, including the Conflict Probe, is the four-dimensional (4D) trajectory synthesis process. As shown in Figure 1, the trajectory synthesis process provides the Conflict Probe with predicted 4D trajectories for every aircraft within the Center airspace. In the CTAS architecture, the number of trajectory synthesis processes producing trajectories can be scaled to meet the requirements for trajectory update rate, based on traffic load in the Center.

The CTAS Conflict Probe consists of a graphical user interface and an underlying conflict detection algorithm. The graphical user interface is contained within the controller display process, and is not the topic of this paper. The Conflict Probe can be used in conjunction with the Traffic Management Advisor (TMA), which employs a dynamic scheduler to meter traffic through metering gates. The conflict detection algorithm and the TMA, while separate software entities, use the same trajectories produced by the trajectory synthesis process. The conflict detection algorithm consists of approximately 30,000 lines of C and C++ code, and is implemented in a CTAS process called PFS_C [6].

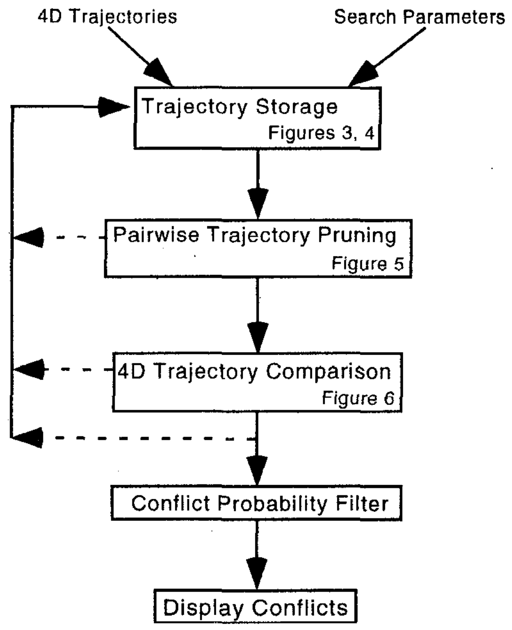


Figure 2. Conflict Detection Algorithm

Figure 2 summarizes the components of the conflict detection algorithm: trajectory processing, pair-wise trajectory pruning, 4D trajectory comparison, and conflict probability filtering. The dashed lines of Figure 2 represent the possible conflict detection cycle exit points for a trajectory pair: thereby avoiding additional computations after it has been determined that pair is not in conflict. The discussion below describes the processing embedded in each block in more detail.

The basic input to the detection algorithm are 4D trajectories, which provide knowledge of future aircraft position. These trajectories are provided by the trajectory synthesis algorithm in CTAS. It uses point mass equations of motion to model vertical and longitudinal accelerations and concatenated segments of straight lines and circular arcs to model horizontal maneuvers and flight paths. A sample 4D trajectory as modeled in CTAS is shown by Figure 3. The times depicted along the trajectory correspond to control points, at which key trajectory characteristics change. For example, the top of descent point (t_3) for the aircraft of Figure 3 corresponds to both a change in speed and descent rate; this is a control point for the trajectory. The problem of synthesizing 4D trajectories and evaluating their prediction accuracy has been examined in a series of reports and papers [5, 7, 8]. The accuracy of CTAS 4D trajectories provides the

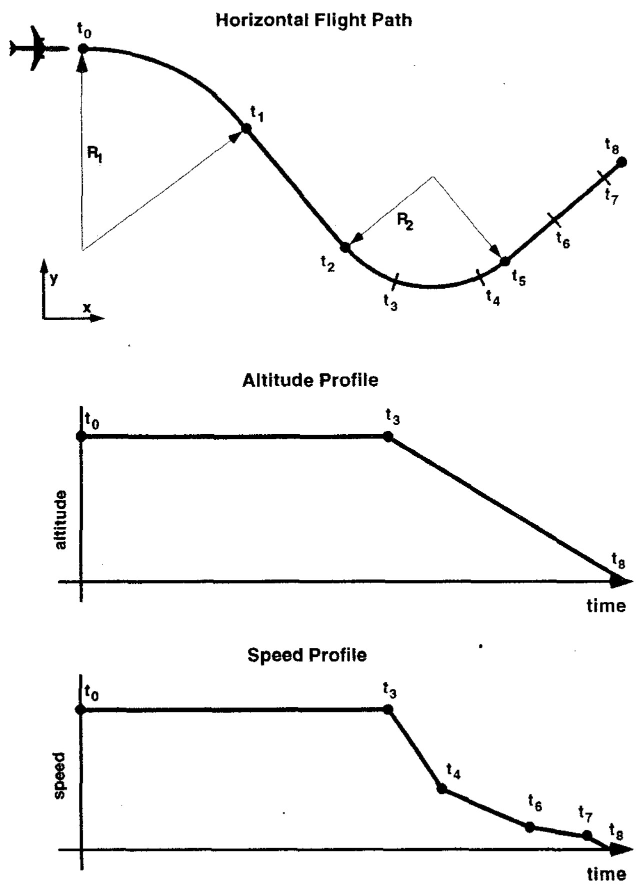


Figure 3. Example of a 4D Trajectory

conflict detection and analysis algorithm with the necessary relation of time and aircraft position to predict conflicts at least 20 minutes into the future.

Trajectory data storage and access is largely determined by the search method selected. Each trajectory comparison could be made by deriving a closed form solution for minimum separation, or it can be made by separation calculations at discrete, closely spaced time instants along the trajectory. A closed form solution for minimum separation is in itself a complex problem, especially when aircraft dynamics and routing have to be considered. The alternative of computing separation at discrete intervals along the trajectory is a simple calculation, but if repeated too often, is also computationally intensive. The conflict search algorithm implemented in CTAS employs discrete comparisons, but with measures taken to limit the number of calculations required.

Trajectory data derived from 4D trajectories, as illustrated in Figure 3, are stored as a series of

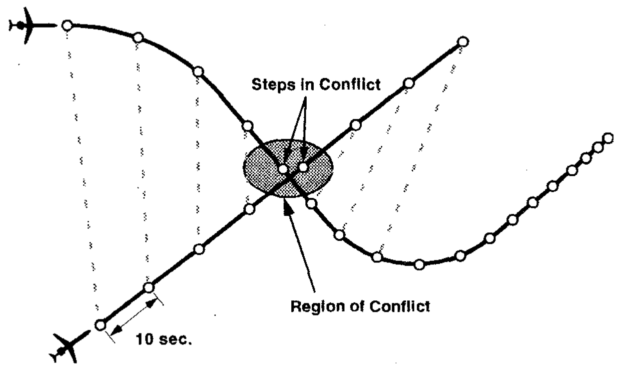


Figure 4. Conflict Detection via Time Step Comparison

aircraft state vectors (x, y, h, V, etc.) at 10 sec. intervals along the flight path. These evenly spaced (in time) state vectors, hereafter referred to as time steps, are compared with time steps of other aircraft to detect potential conflicts. Figure 4 demonstrates the use of trajectory time steps to detect a future conflict, as depicted by the dashed lines; connected time steps correspond to the same time instant, thus forming the basis for separation calculations at each time step.

Some trajectory pairs can be filtered from the conflict search prior to the computationally intensive 4D trajectory comparison. This process, referred to as pair-wise trajectory pruning, is intended to quickly screen all trajectory pairs for those which have no potential for conflict. For example, if a pair of trajectories is separated vertically by at least the minimum required vertical separation throughout the conflict search time horizon, there is no potential for conflict; and thus no need for comparing trajectory time steps for separation. Altitude pruning is accomplished with a simple check of the altitude profiles of the 4D trajectories. As shown in Figure 5, by enclosing each altitude profile in a rectangle, it is easily seen if there is potential for conflict based on vertical separation of the trajectories. The range of possible altitudes, which is needed to form the bounding rectangles in the vertical plane, is obtained from an analysis of the flight plan of each aircraft. Observations indicate 60-80% of all possible trajectory pairs are typically excluded from the 4D search by altitude pair-wise pruning alone. Pruning a trajectory pair based on altitude separation consumes 1/40th of the time necessary to complete a 4D trajectory comparison using 10 sec. time steps. Pruning trajectory pairs based on enclosure of

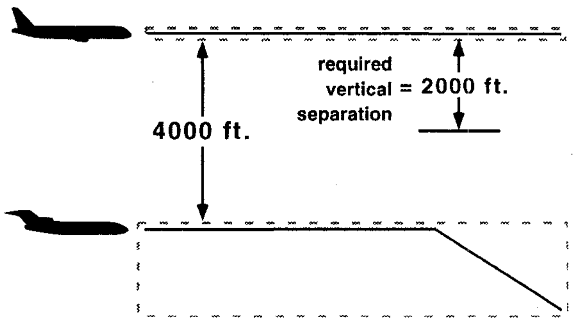


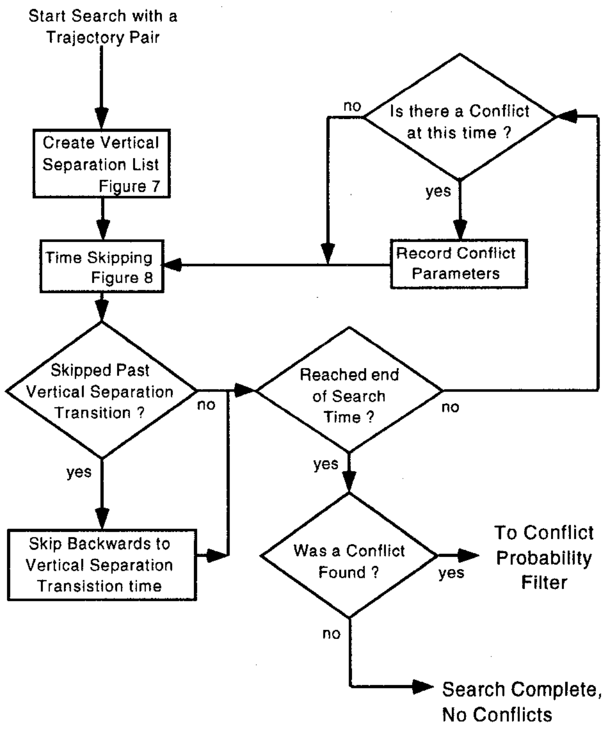
Figure 5. Pair-wise Pruning by Altitude

horizontal flight paths in rectangles was also implemented, but did not demonstrate benefits sufficient to warrant its retention.

4D CONFLICT SEARCH ALGORITHM

If two aircraft trajectories are not pruned due to altitude separation, the individual time steps must be compared to detect potential conflicts. While the number of pairs to compare is considerably reduced, the remaining pairs must be checked by comparing 4D trajectories, a computationally costly process. Three methods are used to reduce the computational load of the 4D trajectory search: time skipping, efficient implementation of variable vertical separation requirements, and avoiding redundant computations. Figure 6 shows the structure of the 4D search algorithm.

Two factors contribute to the need for variable vertical separation requirements for the conflict search algorithm: legal separation requirements and increased uncertainty in trajectory prediction of non-level flight. First, aircraft are legally required to be separated vertically by 1000 ft. below FL290, and by 2000 ft above that level. This dictates that required vertical separation between two aircraft could change if one of the aircraft climbs or descends through FL290 within the conflict detection interval. Additionally, prediction of non-level trajectory segments is less certain than prediction of level trajectory segments. Due to this higher level of uncertainty, controllers may desire conflict detection based on greater than legal vertical separation requirements when one or both aircraft are changing altitude at the time of minimum separation. The Conflict Probe provides the ability to specify three vertical separation criteria in addition to the legally required criteria, depending on the conflict type: 1) both aircraft level flight, 2) one aircraft changing altitude,



Start Search with a Trajectory Pair
Create Vertical Separation List Figure 7
Time Skipping Figure 8
Is there a Conflict at this time ?
no
yes
Record Conflict Parameters
Skipped Past Vertical Separation Transition ?
no
yes
Reached end of Search Time ?
no
yes
Skip Backwards to Vertical Separation Transistion time
Was a Conflict Found ?
yes
To Conflict Probability Filter
no
Search Complete, No Conflicts

Figure 6. 4D Trajectory Search Algorithm

and 3) both aircraft changing altitude. Therefore, it is possible for the required separation between two aircraft trajectories to change a number of times within the detection interval. For example, Figure 7 illustrates the use of increased vertical separation requirements for detection of conflicts in which one aircraft is changing altitude. In this example, the controller has specified the vertical separation required when one aircraft is changing altitude as 3000 ft. as opposed to 1000 ft. when both aircraft are in level flight. This dictates increased separation requirements during the intervals t_0 to t_1 and t_2 to t_3.

It would not be efficient to calculate the required vertical separation at each time step comparison, and it is not required that the vertical separation be calculated for all trajectory pairs: only those that have not been pruned due to altitude separation. A method has been developed which requires a minimal amount of computations for determining required vertical separation at all times for a trajectory comparison.

At the onset of the conflict detection cycle, the flight segment and flight segment transition times are determined for each aircraft trajectory, as well as any altitude transition times, as shown in Figure 7. Because only trajectory pairs can be pruned from

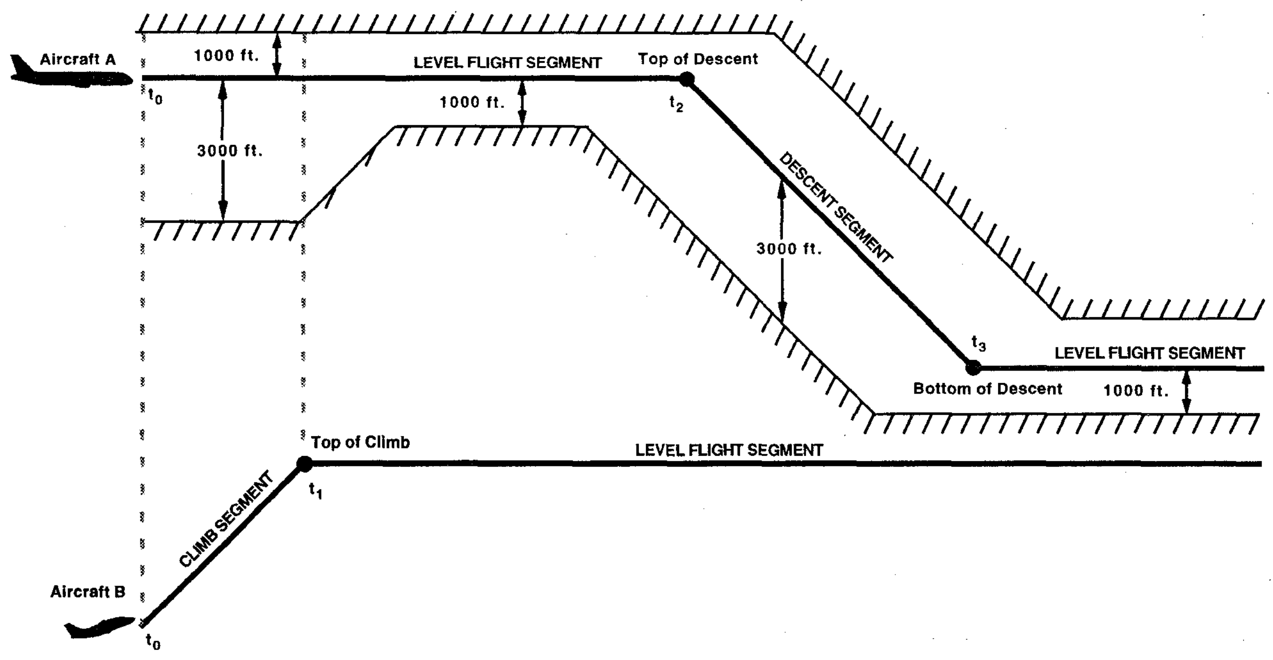


Figure 7. Example of Variable Vertical Separation Requirements

the search, it is necessary that the flight segments be determined for all trajectories at the beginning of a detection cycle. Once it is determined that a 4D search is needed for two trajectories, the required vertical separations must be determined. From the flight segments and altitude transitions for each trajectory, a list can be constructed which defines the transition times between different values of required vertical separation. This list of vertical separation transition times is constructed prior to the 4D search of the trajectories. Therefore, it is only required to check if a transition time has been reached at each comparison to determine the vertical separation, rather than to calculate the flight segments, altitudes, and required separation at each comparison. An extension of this basic model is implemented which effectively models uncertainty in descent and climb portions of the trajectory. The vertical separation required during non-level portions of flight is increased to account for this uncertainty. The onset of this increased separation is gradually phased in to account for the variability of the top of climb and top of descent points, as illustrated in Figure 7.

Time skipping is the process of ignoring portions of the trajectory in the near future where there is no potential for conflict. For example, as shown in Figure 8, two aircraft initially separated by a distance much greater than the required separation should not be checked for separation at every 10 second step along the trajectories; this would be a waste of time considering their actual closure rate. Time skipping is a process of skipping forward in time on the trajectory to a point at which there is potential for conflict.

Time skipping is efficiently accomplished through the trajectory time steps previously described. Because time steps are equally distributed in time along the trajectories, estimates of aircraft performance can be used to determine the first time at which a conflict is possible, and therefore, the time steps that correspond to that time. Skipping too far could result in missed alerts; however, by conservatively selecting maximum aircraft performance, missed alerts are prevented. For example, due to wind modeling errors, a conflict may be missed when time skipping if ground track speeds are used to determine how much time to skip. By using Mach 2 (standard sea-level conditions) as the closure rate, it is unlikely any combination of wind modeling error or encounter geometry would lead to a missed alert. Mach 2 represents an upper limit on closure rate, as it assumes a head-on closure with each aircraft traveling at Mach 1.

A simple design guideline which postpones computations until they are absolutely necessary reduces the number of computations required, and thus the time required to complete the search. For example, to determine if two aircraft are in

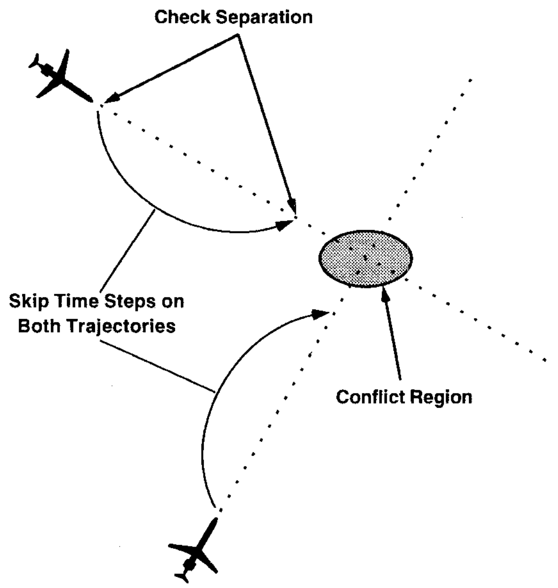


Figure 8. Time Skipping Example

horizontal conflict, it is necessary to compare the sum of the squares of the x and y separations stored in the time steps with the square of the required horizontal separation. However, comparing the sum of the squares is only necessary if both the x and y separations are less than the required horizontal separations. These simple steps reduce computation of the actual separation to once per conflict pair, at the minimum separation. The amount of time saved by this ordering of operations may be small for an individual time step comparison, but when the savings are accumulated over a large number of time step and trajectory comparisons, a substantial increase in search speed is achieved.

Conflict Probability Filter

Uncertainty in conflict detection arises due to wind modeling and prediction error, and partly due to tracking, navigation and control error. Detected future conflicts with high uncertainty are not likely to require evasive action, leading to a false alert if the controller is notified. This is accounted for by a conflict probability filter integrated into the conflict detection and analysis algorithm. The conflict probability algorithm implemented in the Conflict Probe applies to all flight conditions and conflict geometries, not just level flight conflicts [6]. The conflict probability is determined from a statistical model of trajectory prediction error and the encounter geometry at the predicted point of minimum separation [3]. Detected future conflicts with low probability (< 50%) are excluded from the list displayed to the controller, thus limiting the false alert rate. As shown in Figure 6, conflict probability is only calculated for those trajectory pairs predicted to be in conflict. This is justified based on assumptions used to define error models within the conflict probability algorithm. The prediction errors are modeled as normal distributions. Thus, for the case of two aircraft having a predicted minimum separation exactly equal to the required minimum separation, and having independent, normal error distributions, the maximum probability of conflict is 50% [6]. When conflict encounter geometry and other errors are considered, the probability is further reduced below the 50% threshold used by the filter. High probability conflicts (0.85 and larger) are displayed in yellow or red to draw the controller's attention to them [6].

To prevent missed alerts for short time horizons, the conflict probability filter is disabled for conflicts less than 6 minutes into the future. This gives the controller time to recognize, evaluate, and react to a potential conflict even if its probability estimate is less than the cutoff value of the filter. This threshold value of 6 minutes was determined from controller feedback following real-time simulations of the Conflict Probe.

Trial Planning

Trial planning is implemented in the Conflict Probe to assist controllers in resolving conflicts, and to verify that user requests are free of conflict. Three requirements distinguish trial planning from the standard conflict search algorithm: multiple, simultaneous trial plans for a single aircraft, controller interaction, and rapid feedback. Furthermore, the controller may select any aircraft for trial planning, not just those in conflict.

To allow multiple users (e.g. Traffic Management Coordinators or Sector controllers) to perform trial planning for the same aircraft simultaneously, and to allow for more rapid update, the trial plan conflict search is independent of the global conflict search cycle. The primary benefit of an independent search cycle is a large reduction in the number of trajectory pairs to be searched. For n aircraft, a single trial plan requires (n -1) trajectory comparisons rather than the n(n-1)/2 required by the global search; thus reducing the number of

Aircraft in Conflict	Time to Conflict (min)	Altitude Separation (ft x 100)	Horizontal Separation (nmi)	Conflict Probability
AC1 < > AC3	**6**	**0**	**1**	**0.84**
AC1 < - AC2	**12**	**6**	**3**	**0.86**

Cruise Segment **Descent Segment (+ for Climb)**

Figure 9: Example of a Conflict List

comparisons required for a reasonable number of trial plans.

Trial planning is used interactively by the controller to resolve conflicts. For example, in an attempt to vector an aircraft to avoid a detected conflict, a controller may stretch the flight path of a trial plan trajectory, continual monitor the conflict status, and then accept the trial plan once the conflict has been resolved. The controller should perceive the time required to determine if a proposed resolution is conflict free as nearly instantaneous. A one second update rate was selected as the update rate for the trial plan search algorithm as an achievable update rate that allows controller interaction with small but acceptable lag.

RESULTS

The output of the conflict prediction and analysis algorithm is a list of potential conflicts displayed to the sector controller. Upon receipt of the list of conflicts, the controller can take one of two possible actions: 1) initiate resolution maneuvers by issuing a clearance, or 2) delay the resolution maneuvers pending results of the next conflict cycle update. Effective use of the first option relies on presenting the controller with the necessary conflict information to propose a resolution, as well as the means to do so in a timely manner. If a controller chooses to postpone a resolution decision, it is essential to provide the controller with the latest conflict prediction information at an update rate equivalent to that of the radar update cycle of 12 sec. An update rate that matches that of the radar ensures flexibility, timeliness and accuracy of the predicted conflict information as previously discussed. By using the latest available radar tracking information and trajectory predictions, the conflict list is always up to date, thereby increasing its usefulness to the controller.

As shown in Figure 9, the conflict list provides the controller with the following information for each aircraft pair in conflict: 1) the aircraft callsigns, 2) the segment of flight of each aircraft at the time of the conflict, indicated by the symbols <, -, and +, representing cruise, descent, and climb segment conflicts, 3) the time until first loss of required separation, 4) the minimum predicted altitude separation, 5) the minimum predicted horizontal separation, and 6) the conflict probability (optional).

Based on this information, the controller can propose a resolution using the trial planning functionality previously discussed, receiving nearly instantaneous conflict prediction feedback of the trial resolution. With an appropriate computer human interface (CHI), the sector controller can resolve future conflicts at an earlier time than is currently possible.

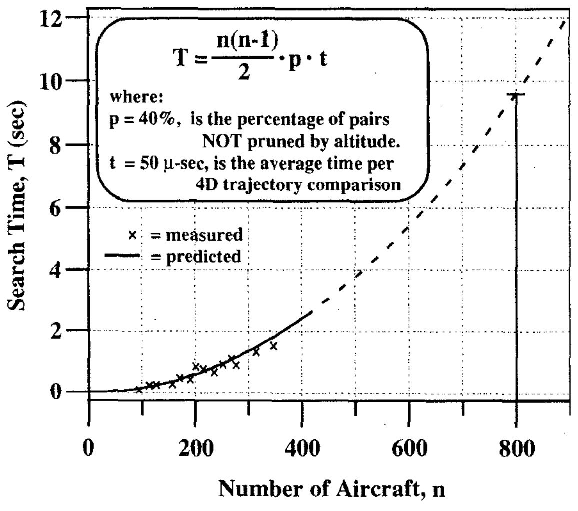


Figure 10: Conflict Search Performance

The computational performance of the conflict search algorithm was evaluated on a Sun Microsystems Ultra workstation (340 MIPS).

Performance was evaluated with live traffic received over a high speed data link from the Denver Air Route Traffic Control Center (ARTCC). Figure 10 shows typical performance of the conflict detection and analysis algorithm. It indicates that over 800 aircraft can be processed in less than 10 seconds, a number in excess of the expected maximum for two ARTCCs during peak traffic periods, and well in excess of that required for single ARTCC operation.

CONCLUDING REMARKS

A conflict detection algorithm has been designed and implemented in CTAS. The algorithm identifies those aircraft pairs which are in potential future conflict, as well as providing important conflict characteristics such as time to loss of separation, its likelihood of occurrence, and conflict encounter geometry.

The detection algorithm is based on analysis and pair-wise comparison of 4D trajectories. It employs pair-wise trajectory pruning by altitude to reduce the number of trajectory pairs that must be evaluated by the computationally intensive 4D search algorithm. Approximately 60-80% of all trajectory pairs are eliminated from the 4D search through these pruning heuristics.

Where altitude pruning cannot rule out the possibility of a conflict, the algorithm uses discrete steps in time along the 4D trajectories to detect conflicts. Three methods are employed to decrease the computations required to complete the 4D search: 1) time skipping when aircraft are separated by large distances, 2) efficient implementation of the requirement for multiple vertical separation criteria, and 3) ordering of separation calculations so as to avoid redundant and complex calculations. Results indicate more than 800 aircraft can be processed in a period of 10 sec. This suggests that the search algorithm easily meets the performance requirements for an automated conflict detection and resolution tool in the current air traffic system.

To reduce the number of false alerts, and to provide controllers with information related to the certainty of a conflict, the analysis algorithm employs conflict probability estimates. The resulting list of conflicts can then be used by the sector controller to resolve conflicts as much as 15 minutes earlier than is possible in the current system. The conflict detection algorithm is integrated with a trial planning function used by controllers to resolve conflicts. Conflict detection for trial planning is performed once per second for each trial plan aircraft, providing near instantaneous response to controller input.

Favorable controller evaluation of the CTAS Conflict Probe through real-time simulation has led to the decision to evaluate the system in a limited operational environment. The Conflict Probe has been adapted for the Denver and Fort Worth Centers, and is scheduled for field evaluation beginning in September, 1997.

REFERENCES

[1] Final Report of the Task Force 3, Free Flight Implementation. RTCA Inc., Washington, DC, Oct 26, 1995. Available from RTCA at 202-833-9339.

[2] "Free-For-All Flights," *Scientific American*, vol. 273, no. 6, pp. 34-37, Dec, 1995.

[3] Paielli, R.A.; Erzberger, H.: "Conflict Probability Estimation For Free Flight," J. Guidance, Control, and Dynamics, vol. 20, no.3, pp.588-596, May-June 1997.

[4] Erzberger, H.; Davis, T.J.; Green, S.: "Design of the Center-TRACON Automation System," AGARD Guidance and Control Symposium on Machine Intelligence in Air Traffic Management, Berlin, Germany, May 1993.

[5] Slattery, R.; Zhao, Y.: "Trajectory Synthesis For Air Traffic Automation," AIAA J. Guidance, Control, and Dynamics, vol. 20, no. 2, pp. 232-238, March-April, 1997.

[6] Erzberger, H.; Paielli, R.A.; Isaacson, D.R.; Eshow, M.M.: "Conflict Detection and Resolution in the Presence of Prediction Error," 1st USA/Europe Air Traffic Management R&D Seminar, Saclay, France, June 17-20, 1997.

[7] Erzberger, H.; Tobias, L.: "A Time Based Concept For Terminal Area Traffic Management," NASA Technical Memorandum TM-88243, April 1986.

[8] Green, S.; Vivona, R.: "Field Evaluation of Descent Advisor Trajectory Prediction Accuracy," AIAA Guidance, Navigation, and Control Conference, San Diego, CA, July 29-31, 1996.