Conflict Detection and Resolution In the Presence of Prediction Error

Heinz Erzberger,* Russell A. Paielli,† Douglas R. Isaacson,‡ and Michelle M. Eshow§
NASA Ames Research Center, Moffett Field, CA 94035-1000

Abstract

This paper describes the design of a conflict detection and resolution tool, referred to as a Conflict Probe, for use by enroute (Center) air traffic controllers. This tool is designed to help detect and resolve all classes of conflicts, such as overflight/overflight, arrival/overflight, climbout/overflight, etc., up to twenty minutes in advance. The design is based on an approach that combines deterministic trajectory prediction and stochastic conflict analysis to achieve reliable conflict detection. The paper begins by formulating error models for trajectory prediction. Then an efficient algorithm is described for estimating conflict probability as a function of encounter geometry. The conflict probability theory and algorithm are applied to the design of the Conflict Probe and are further used to develop a method of automated conflict resolution with constraints on the post-resolution conflict probability. Next, aircraft performance models and equations of motion for generating four-dimensional trajectories are presented. The paper concludes with a description of the methods used to minimize the time to search the trajectories for potential conflicts. Performance tests of the search algorithm indicate that up to 800 aircraft can be processed by the Probe within a search cycle of 10 seconds. The Conflict Probe has been implemented as a software process within the Center-TRACON Automation System (CTAS). It uses the trajectory synthesis capability of CTAS to generate predicted trajectories for the conflict search and trial resolution processes. Field evaluation of the Probe will be conducted at the Denver and Fort Worth Centers beginning in September 1997.

*Senior Scientist for Air Traffic Management, Flight Management and Human Factors Division (AF Division), mailstop 210-9, herzberger@mail.arc.nasa.gov
†Aerospace Engineer, AFC Branch, mailstop 262-3
‡Aerospace Engineer, AFA Branch, mailstop 210-9
§Aerospace Engineer, AFD Branch, mailstop 210-9
[0]This paper was prepared for the 1st USA/Europe Air Traffic Management R&D Seminar, Saclay, France, June 17-20, 1997

Introduction

The efficiency of air transportation in the U.S. could be improved significantly if the fixed jet airway network for routing airline flights were relaxed to allow more direct or wind-optimal trajectories. While the current network of jet routes helps to maintain the safe and orderly flow of traffic, new technologies for navigation, communication, and automation will enable the relaxation of routing restrictions inherent in the current system without compromising safety. The ultimate goal is "Free Flight" [1, 2], which would essentially eliminate the fixed network of jet routes everywhere except in terminal areas surrounding large airports.

Automated conflict prediction and resolution designed to work at least ten minutes before a conflict can occur is considered a basic requirement for achieving "Free Flight." The basic problem of conflict prediction is inherent in the nature of trajectory prediction; namely, that errors in prediction are unavoidable. The farther in the future a prediction is made, furthermore, the greater the probability of error. A method is needed, therefore, to estimate the magnitude of the prediction error and its effect on the probability of conflict. A conflict is defined as two or more aircraft coming within the minimum allowed distance and altitude separation of each other. The minimum allowed horizontal separation for enroute airspace is currently 5 nautical miles (nmi). The vertical separation requirement above flight level 290 (29,000 feet (ft)) is currently 2000 ft; below that level it is 1000 ft.

The optimal time to initiate a conflict resolution maneuver is a trade-off between efficiency and certainty. The farther in advance a resolution maneuver is initiated, the more efficient it is likely to be in terms of extra time and distance flown, but the less certain will be exactly what maneuver is required or whether a maneuver is required at all. The later a maneuver is initiated, on the other hand, the more certain will be exactly what maneuver is required, but the larger and less efficient the maneuver is likely to be. Knowledge of conflict probability can help to establish the optimal time to initiate a resolution maneuver as well as the charactistics of the

maneuver.

The objective of an early resolution maneuver is to reduce the post-resolution conflict probability to a small but non-zero value. It is neither desirable nor possible to reduce the post-resolution conflict probability to zero when the time to go to the predicted conflict is larger than a few minutes. The intent is to help controllers detect and resolve likely conflicts earlier than they can today, not to eliminate the need for human controllers, who will continue to have ultimate responsibility for ensuring proper separation at all times.

The conflict detection process begins by calculating four-dimensional (4D) predicted trajectories for every radar-tracked aircraft in a region of airspace. A 4D trajectory incorporates all information that is currently known about an aircraft, such as its current state and its intent, at the time of computation and therefore constitutes the best estimate of the future 4D position of that aircraft. The complex modeling, algorithmic and software processes involved in computing 4D trajectories comprise the core element of the Center-TRACON Automation System (CTAS) [3]. The Conflict Probe described in this paper utilizes the trajectory prediction system of CTAS and is embedded in the architecture and software of CTAS.

This paper is comprised of four main sections which together describe the essential design features of the Conflict Probe. The first section gives an abbreviated derivation of the conflict probability algorithm recently published in [4]. It also describes its extension to the 3 dimensional case and discusses several example results. The next two sections focus on applying the conflict probability algorithm to filter out low probability conflicts and on applying it to perform conflict resolution. Then an overview is given of the software and hardware architecture of CTAS. It includes a summary of the analytical models and computational algorithms used by CTAS to generate 4D trajectories. The last section describes the algorithm that searches the 4D trajectories for potential conflicts. It shows how the software has been designed to perform the conflict search efficiently for a large number of aircraft, within a short time interval.

The incorporation of a Conflict Probe in CTAS enhances the performance of existing CTAS tools as well as expanding the coverage of CTAS to the whole of Center airspace. The Conflict Probe is needed to support the operation of the Descent Advisor (DA), a controller tool for managing arrival traffic in Center airspace. As a standalone tool, referred to as the User Preferred Routing (UPR) Conflict Probe, it will enable controllers to detect and resolve conflicts between aircraft on non-standard routes earlier, thereby reducing routing restrictions in Center airspace and contributing to the achievement of Free Flight.

Conflict Probability Estimation

The conflict probability algorithm presented in [4], which applied only to level flight, has been generalized to apply to non-level flight. That generalized algorithm is summarized in this section. A more detailed discussion will be presented in a future paper.

Trajectory prediction is inexact, primarily because of wind modeling and prediction error and secondarily because of tracking, navigation and control error. The conflict probability algorithm requires a prediction of positions and velocities of a pair of aircraft at their point of minimum separation (provided by CTAS) and a statistical model of the prediction errors.

Prediction errors can be represented as ellipses in the horizontal plane or as ellipsoids in three-dimensional space. Those error ellipsoids tend to have their principal axes in the along-track, cross-track, and vertical directions. Since aircraft are usually flown at a constant airspeed or Mach number rather than a constant groundspeed, the effects of wind modeling and prediction errors accumulate with time, particularly in the along-track direction. Figure 1 shows a typical example of the growth of two-dimensional (2D) prediction error ellipses in the along-track direction.

The trajectory prediction error for an aircraft will be modeled as normally distributed [5], with zero mean and with a covariance that has eigenvectors in the along-track, cross-track, and vertical directions, as explained previously. The covariance matrix is therefore diagonal in a coordinate system aligned with the aircraft heading. If S is the diagonal covariance matrix, and R is the rotation matrix that transforms the heading-aligned coordinates to the reference coordinates, then the transformed covariance matrix is

$$Q = RSR^T \tag{1}$$

Combined Covariance

Because the trajectory prediction errors are modeled as normally distributed, the two error covariances for an aircraft pair can be combined into a single equivalent covariance of the position difference or the relative position of one aircraft with respect to the other. For present purposes, this combined covariance can be assigned to one of the aircraft, referred to as the "stochastic" aircraft, and the other aircraft, referred to as the "reference" aircraft, can be regarded as having no position uncertainty.

Subscripts S and R will be used to designate the stochastic and reference aircraft, respectively. The combined prediction error covariance is then

$$M = Q_S + Q_R - Q_{SR} \tag{2}$$

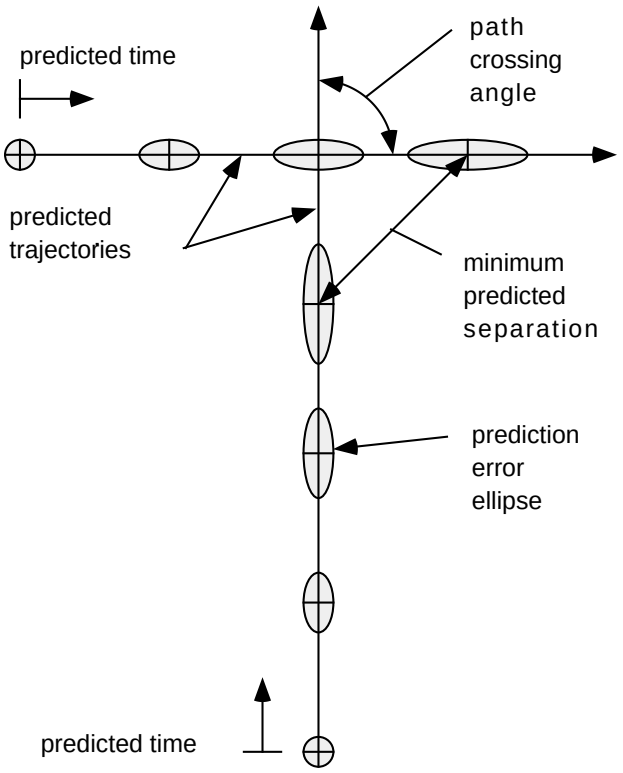

Figure 1: Trajectory prediction error ellipses

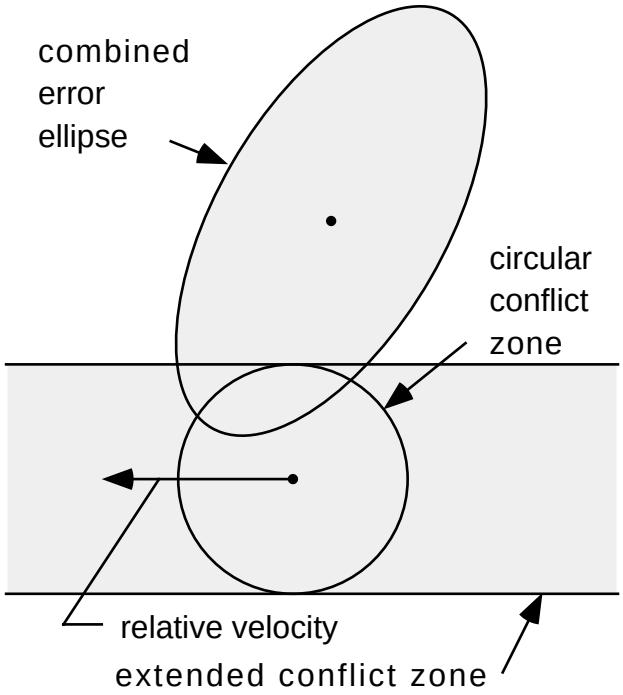

Figure 2: 2D encounter geometry

where Q_S and Q_R are the individual covariances based on equation 1, and Q_{SR} is the cross-correlation term. The cross-correlation term accounts for the fact that the wind modeling error is spatially correlated and a portion of its effect cancels in the position difference. In general, the combined error ellipsoid corresponding to M will no longer have principal axes aligned with the along-track and cross-track directions of either aircraft.

Figure 2 shows an example 2D encounter geometry in the horizontal plane; the three-dimensional (3D) case is more difficult to illustrate but similar in principle. The combined error ellipse is centered on the stochastic aircraft, and the circular conflict zone (nominal 5 nmi radius) is centered on the reference aircraft. The error ellipse corresponds to a probability density function that can be represented as a surface over the ellipse, where the total volume under the surface is unity. The probability of conflict at a *particular time* is the portion of that volume that is within the circular conflict zone, but this probability is not as important as the *total* probability of conflict for the encounter, which is discussed in the following paragraphs.

It is assumed that the aircraft velocities and prediction errors are constant during the encounter or period of potential conflict, which will be at least approximately true for most aircraft pairs in free flight. Without these assumptions, an analytical solution is much more difficult or perhaps impossible to find. Note that prediction errors due to planned turns or other maneuvers that will be completed before the encounter begins can be properly accounted for in the covariance matrices.

The total probability of conflict for the encounter can then be determined as follows. Project the conflict zone along a line parallel to the relative velocity to form an extended conflict zone, as illustrated in figure 2 for the 2D case. The conflict probability is equal to the portion of the volume under the probability density surface that is within this extended conflict zone. The coordinate transformation to be presented in the following paragraphs allows this probability to be determined analytically.

Coordinate Transformation

The conflict probability is difficult or impossible to determine analytically in the original coordinate system. It can be determined numerically, but a numerical solution is likely to be less accurate and much less efficient than an analytical solution. Such inefficiency is undesirable for an algorithm that is intended to run in real time for extended periods of time. Fortunately, a coordinate transformation has been found that allows an exact analytical solution for the case of level flight and a good

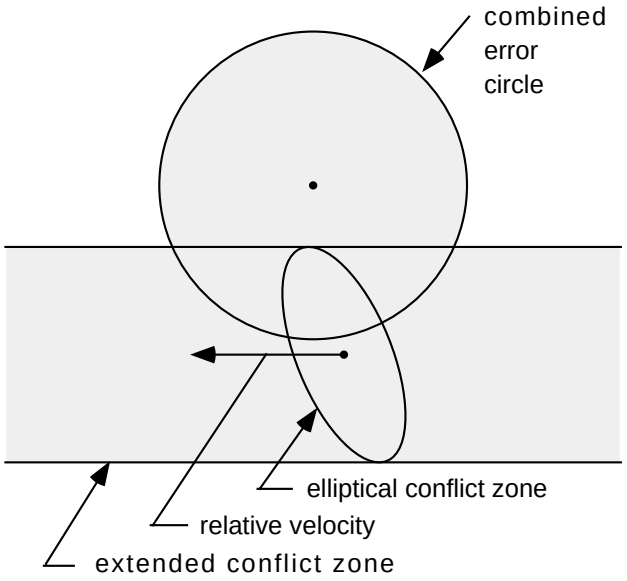

Figure 3: Transformed 2D encounter geometry

analytical approximation for the case of non-level flight.

The coordinate transformation consists of a nonorthogonal transformation followed by an orthogonal transformation (rotation), then a projection onto a plane. The nonorthogonal transformation transforms the combined error ellipse into the standard form of a unit sphere. The orthogonal transformation aligns the relative velocity with one of the coordinate axes. The projection projects the cylindrical conflict zone onto the plane normal to the relative velocity. These coordinate transformations must be applied to the relative positions and velocities of the two aircraft, the combined error ellipse, and the cylindrical conflict zone.

If T is a coordinate transformation matrix, and M is the covariance matrix in the original coordinate system from equation 2, then the combined error covariance in the transformed coordinate system is TMT^T. A Cholesky decomposition or "square-root" factorization of M is of the form

$$M = LL^T \tag{3}$$

where L is lower triangular. If the transformation matrix T is selected such that

$$T = L^{-1} \tag{4}$$

then the transformed covariance matrix is

$$TMT^T = I \tag{5}$$

where I is an identity matrix. The combined error ellipsoid is then transformed into the standard form of a unit sphere. Figure 3 illustrates an example of transformed encounter geometry for the 2D case; the 3D case is similar in principle.

In the original coordinate system, the cylindrical conflict zone is represented by the horizontal circles at the top and botton of the cylinder. In the new coordinate system, those circles become ellipses (ellipsoids of zero thickness) that remain in the horizontal plane for level flight but not necessarily for non-level flight. The conflict zone, which was a circular cylinder in the original coordinate system, is therefore an elliptical cylinder in the transformed coordinate system. The details of these transformations will be presented in a future paper.

Having the error ellipsoid in the form of a unit sphere simplifies the probability computation considerably because the corresponding 3D probability density function decouples into the product of three identical one-dimensional functions. For the 2D case, the probability density function can be represented as a radially symmetric surface over the error circle, where the total volume under the surface is unity. For the 3D case, the probability density function can be represented as a radially symmetric mass distribution, where the total mass is unity.

In the transformed coordinate system, the extended conflict zone is still in the direction of the (transformed) relative velocity. For the 2D case the conflict probability is still equal to the portion of the volume under the probability density surface that is within this extended conflict zone. An orthogonal transformation can be used to rotate the transformed coordinate system such that the relative velocity is in the x direction. Note that a rotation will not change the shape of the error sphere. The rotation consists of two steps: first, a rotation about the z axis such that the relative velocity is in the x-z plane; second, a rotation about the y axis such that the relative velocity in the x direction.

An example of transformed 3D encounter geometry is illustrated in figure 4. The extended conflict zone is the projection of the cylindrical conflict zone in the direction of the relative velocity (the x direction, which is perpendicular to the paper in figure 4). The view of the cylindrical conflict zone from the direction of the relative velocity is therefore the cross-section of the extended conflict zone. The cylindrical conflict zone appears very distorted in the transformed coordinates because the combined vertical uncertainty ($\approx$ 100 ft) is usually much less than the horizontal uncertainty, so the height of the cylinder gets magnified in the transformation (the distortion is actually worse than shown in the figure). The vertical velocity also gets magnified. (Note that the allowed cruise altitudes in Free Flight may be in finer increments than the current increments of 1000 ft.)

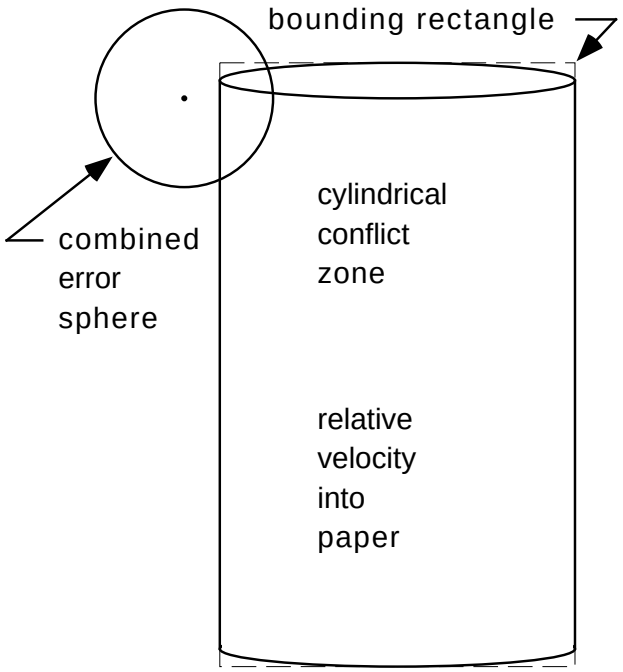

Figure 4: Transformed 3D encounter geometry

For the important special case of level flight, the cylindrical conflict zone appears as a rectangle when viewed horizontally from the direction of the relative velocity. The conflict probability then decouples into the product of vertical and horizontal conflict probabilities and can be determined exactly by the analytical methods presented previously [4]. For nonlevel flight, on the other hand, an exact analytical solution has not been found. Numerical integration could be used, but that would be numerically intensive, which is undesirable for an algorithm that is intended to run in real time for long periods. Fortunately, a very good analytical approximation can be determined by using a bounding rectangle around the cylindrical conflict zone, as illustrated in figure 4.

The next step is to project the conflict-zone ellipses onto a plane normal to the relative velocity. The coordinate transformation was defined such that the transformed relative velocity is in the x direction, so the ellipses must be projected onto the y-z plane. The derivation of this projection is algebraically very complex (it will be left as an exercize for the interested reader), but the result is very simple. The projection onto the y-z plane is achieved by simply taking the y-z components of the vector to the center of the conflict-zone ellipses and the lower-right 2×2 y-z submatrix of the representative matrix.

A minimum bounding rectangle tangent to the projection is then determined. Again, the details will be presented in a future paper. Because the probability density function decouples and the boundaries of the bounding rectangle are parallel to the coordinate axes, the conflict probability estimate simplifies to the product of two Gaussian cumulative differences. This solution can be computed analytically.

Numerical Examples

A set of four plots were generated to show typical conflict probabilities as functions of time to minimum separation. The aircraft speeds were 480 kts in each case, except as noted. The conflict separation distance was 5 nmi, the legally required value for enroute airspace. The cross-track rms error was 1 nmi, and the along-track rms error started at zero and grew linearly at a rate of 15 kts, unless otherwise stated. These values are typical for cruise. (This linear-growth model is typical but is not assumed or required by the algorithm.) Cross correlation of wind errors was not modeled, but could be added without changing the algorithm. The complex nonlinear behavior of the conflict probabability function revealed by these examples underscores the need to incorporate the algorithm into the Conflict Probe.

Figure 5 shows the effect of minimum predicted separation on conflict probability. Conflict probability is plotted as a function of the time to minimum predicted separation, with the minimum predicted separation as a parameter, where the path-crossing angle is 90 deg. For small prediction times, the covariances are small and the conflict probabilities are a strong function of minimum predicted separation, ranging all the way from zero to one. For larger prediction times, the covariances grow and the conflict probability becomes a weaker function of the minimum predicted separation. The conflict probabilities converge and asymptotically approach zero as prediction time increases.

Note that while the conflict probabilities decrease monotonically for minimum predicted separations of 0, 2.5, and 5 nmi, they first increase to a maximum before beginning to converge toward zero for minimum predicted separations of 7.5 and 10 nmi. This behavior reflects the fact that for prediction times up to about 25 min the effect of random prediction errors increase opportunities for conflict more rapidly than they decrease it if the predicted separation is larger than the allowed legal separation of 5 nmi. Note that for an exact predicted collision, the probability of loss of legal separation (5 nmi) decreases to 0.5 at a prediction time of about 30 min.

Figure 6 shows the effect of path-crossing angle on conflict probability. Conflict probability is plotted again as a function of the time to minimum predicted separa-

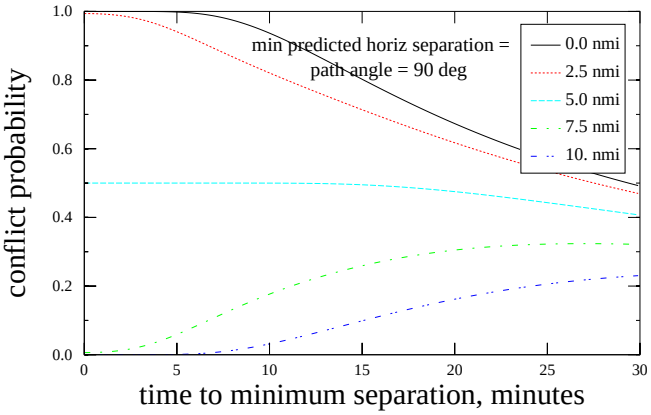

Figure 5: Effect of minimum predicted separation

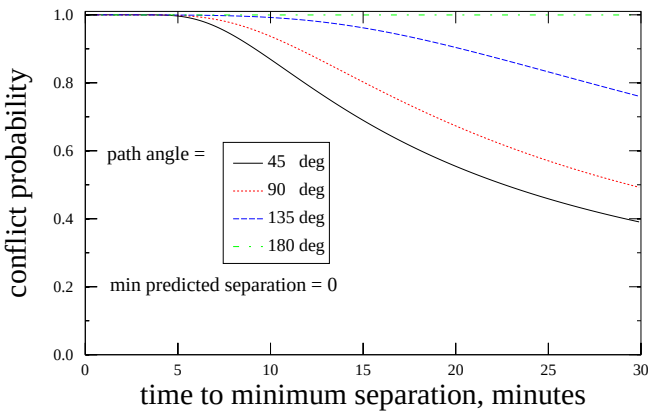

Figure 6: Effect of path crossing angle

tion, but with the path-crossing angle as a parameter, where the predicted minimum separation is 0 nmi. As a point of reference, the curve for the path-crossing angle of 90 deg is identical to the corresponding curve of figure 5. As the prediction time increases, the conflict probability decreases faster for smaller path-crossing angles than for larger ones. As the path crossing angle approaches 180 deg (a head-on encounter), the conflict probabilities remain high and almost independent of prediction time. This behavior reflects the fact that for a head-on encounter with small predicted separation and a constant rms cross-track error a collision is a near certainty regardless of along-track prediction arror.

If cross correlation of wind modeling errors exist and are taken into account, however, these curves could be significantly different for path-crossing angles less than about 45 deg. In that case, a portion of the trajectory-prediction error would cancel in the position difference, and the effective error growth rate would be smaller. Hence the conflict probabilities for such smaller path angles would not decrease as rapidly as shown in figure 6.

Figure 7 shows the effect of constant cruise speed differences for a small (15 deg) encounter angle. All the curves are based on a predicted collision at constant altitude. One aircraft flies at 500 kts while the speed of the other aircraft is reduced in steps of 50 kts down to 300 kts. The figure shows that conflict probability decreases less rapidly with prediction time for larger speed differences. This behavior can be understood by visualizing the encounter from the perspective of an observer riding on one of the aircraft. Having a large difference in velocity at a shallow angle is similar to the case in which a faster aircraft overtakes a slower one on a converging path, diminishing the effect of random along-track errors.

Figure 8 is an example of a cruise vs. descent encounter where one aircraft flies at constant altitude and the other flies first at constant altitude and then begins a descent segment. The encounters occur during the descent, with a minimum predicted horizontal separation of zero and a path crossing angle of 90 deg. The predicted altitude separation at minimum horizontal separation is varied from 0 to 4000 ft in increments of 1000 ft. Both aircraft fly at 480 kts. The descending aircraft descends at a nominal descent angle of 2.5 deg. During cruise the along-track rms error growth rate is 15 kts for each aircraft. During descent the along-track rms error rate increases to 20 kts and the error in the descent rate is assumed to have a constant rms value of 500 ft/min. The predicted minimum horizontal separation occurs 7 min into the descent.

The two-segment altitude profile creates a discontinuity in the slopes of the conflict probability curves at the 7-min prediction time. The curves fall off more rapidly prior to this time and less rapidly thereafter, reflecting the effect of the higher error rates during descent. For the collision case, the conflict probability drops to 0.7 at a prediction time of only approximately 5 or 6 min. The rapid drop-off illustrates the difficulty of accurately predicting conflicts when at least one aircraft is in descent.

Since computational efficiency is a major concern in a real-time air traffic control system, basic timing tests were performed on the conflict probability algorithm running on a Sun SPARC 20 workstation. These tests were for the conflict probability algorithm only and did not include trajectory prediction, wind modeling, or any other part of the problem. The average computation

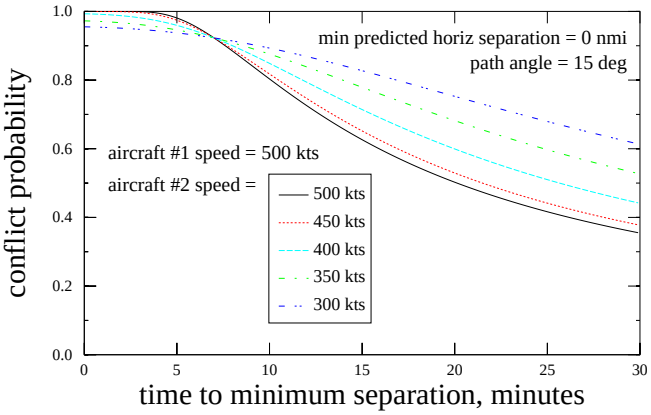

Figure 7: Effect of speed difference

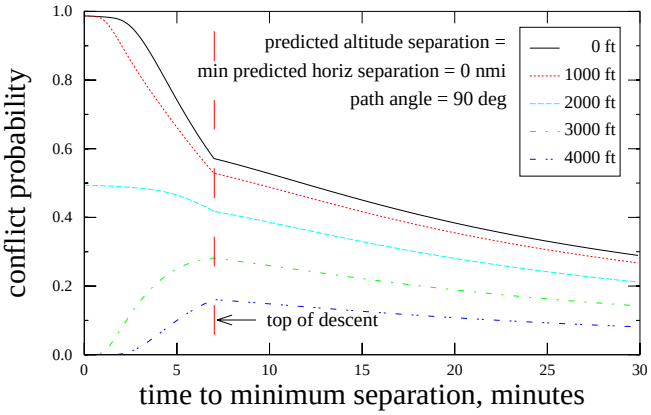

Figure 8: Effect of altitude difference, cruise vs. descent

time per aircraft pair was approximately 1 millisecond. This time is two to five orders of magnitude faster than a numerical solution, depending on the method and level of resolution of the numerical integration. Furthermore, it is fast enough to be used directly in a real-time system.

The conflict probability algorithm is programmed in approximately 5000 lines of C++ code and is integrated into the conflict search and trial resolution process of CTAS. It is used in several ways to enhance the detection process as described in the next section.

Probability Filter for Conflict Analysis

Conflict detection consists of identifying all pairs of aircraft whose distance and altitude separations are predicted to be less than specified minimum values within the detection time horizon. A computationally efficient method for performing this task is described in a later section of this paper. In this deterministic phase of the conflict search, the calculated 4D trajectories used in the search are treated as if they predict the future position and altitude of each aircraft exactly, without any errors. The result of the deterministic search is an initial conflict list, containing the pairs of aircraft that were found to violate the separation criteria. The next step in the conflict search is to incorporate the conflict probability analysis formulated in the preceding section.

In the UPR/DA Conflict Probe, conflict probabilities are not normally shown as numerical values on the controller's conflict list (unless specifically requested), but are indicated instead by a color code. A pair in the conflict list is colored red if the probability that the legally required separation (5 nmi horizontally and 2000 ft vertically) will be violated is greater than a threshold level of $P_{lr} \approx 0.85$. It is colored yellow if the probability that a separation specified by the controller will be violated is greater than a threshold level of $P_{cy} \approx 0.85$. It should be noted that in UPR the controller may specify a separation that is larger than is required legally; for example, if legal separation is 5 nmi. the controller my choose 8 nmi as the separation that triggers display of a conflict pair in the list. A conflict pair is shown in white if its conflict probability is less than P_{cy}. Finally, a conflict pair is not displayed in the conflict list if the conflict probability calculated for the controller-specified separation is less than P_{min}, where P_{min} is in the range of 0.4-0.6, with a nominal value of 0.45. The interval between P_{min} and P_{cy} provides for the display of those conflict pairs which have reasonable expectation to become candidates for resolution in the near future. Awareness of such conflicts gives controllers additional time to formulate strategies for resolution. The values for P_{cy} and P_{lr} and their associated color codes (white, yellow and red) act as decision and priority indicators for controllers to initiate resolution. A value of 0.85 for these variables implies a false alert rate of 0.15, a value that is suggested by the resolution analysis described in the next section.

Prior to computing the conflict probability, the initial state vector for each aircraft of a conflict pair is checked to determine if it satisfies an eligibility condition for a valid probability calculation. The condition consists of checking if the initial state vector lies within specified maximum error tolerances of the planned trajectory derived from the flight plan. The check is performed on the tracking error components defined as the

scalar differences between components of the aircraft and the planned trajectory state vectors at the time of calculation. The tracking errors included in this check are cross-track position error, heading error, altitude error, and altitude-rate error. Only the first three errors are used in the check if an aircraft is at or near its assigned altitude. The altitude-rate error replaces altitude error for an aircraft climbing or descending to a specified (cleared) altitude. Shadow testing of the Conflict Probe with live traffic indicates that acceptable choices for the maximum errors are ±7 nmi, ±25 deg, ±500 ft, and ±500 ft/min, respectively.

If the eligibility conditions are not satisfied, the conflict probability algorithm is not applicable and is therefore not called by the conflict detection process. Failure to meet the eligibility conditions also implies that the intent of the aircraft cannot be reliably inferred from its flight plan at the current time. In this case, the conflict pair is excluded from the conflict list displayed to the controller, but only if time-to-go to the loss of predicted minimum separation is greater than a critical value, t_{pf}, which is in the range of 6-8 min. However, for times-to-go less than t_{pf}, the conflict pair is included in the controller conflict list, even if the eligibility condition is not met. This strategy is based on the observation that as prediction time declines below about 6 min the conflict problem transitions from a strategic problem where close conformance with intent is required for reliability, to an essentially tactical problem where conflicts are determined from current state only and not the flight plan. Ideally, the Conflict Probe should provide a seamless transition between strategic conflict detection with time horizons of at least 20 min to tactical conflict detection with time horizons in the range of 1-2 min.

Conflict Resolution

In the preceding section conflict probability was used as a filter to screen potential conflict pairs before displaying them in the controller's conflict list. Furthermore, conflict pairs were color-coded in the conflict list to indicate high levels of conflict probability, thereby identifying those pairs that controllers should give priority for issuing resolution clearances. This section shows how the synthesis of 4D resolution trajectories can be combined with conflict probability analysis to provide a rational solution to the conflict resolution problem.

Conflict resolution divides into two principal subproblems: (a) deciding the time to initiate a resolution maneuver and (b) selecting the maneuver to execute. The optimal time to initiate a maneuver is a tradeoff between efficiency and certainty. The earlier a maneuver is initiated, the more efficient it is likely to be in terms of extra distance flown, but the less certain it is what maneuver is required or whether a maneuver is required at all. The later a maneuver is initiated, on the other hand, the more certain it is what maneuver is required, but the less efficient and the more severe the maneuver may have to be.

The earliest time that it is appropriate to issue a resolution is determined by specifying $P_{r,min}$, which the conflict probability must equal or exceed before a resolution advisory is issued. The value of $P_{r,min}$ is fixed by the maximum acceptable false alert probability, $P_{fa} = 1 - P_{r,min}$, which lies in the range of 0.1-0.2, as will be explained later.

For each conflict situation, there exist a potentially unlimited number of different 4D trajectories that can resolve the conflict. Fortunately, only the minimum separation distance achieved by a resolution trajectory is needed for conflict resolution analysis in the horizontal plane. If the resolution also involves a vertical maneuver, the minimum predicted vertical separation is also required and would add a second parameter.

Consider first the pre-resolution conflict probability, $P(d_{min} \mid d_l, t_c)$, where d_{min} is the predicted minimum separation just prior to the resolution, d_l is the specified legal (or, alternatively, the controller specified) minimum separation (assumed to be 5 nmi in the example to be discussed), and t_c is time to go to the point of minimum separation. The function P also depends implicitly on numerous other parameters, such as route crossing angle, aircraft velocities and prediction error growth rates; but these assume fixed values in a specific encounter and are therefore irrelevant to this discussion.

Now select any horizontal resolution trajectory that increases the predicted minimum separation from d_{min} to $d_{r,min}$ and changes the time to go from t_c to t_{rc}. Then the post-resolution conflict probability can be written as $P_r(d_{r,min} \mid d_l, t_{rc})$, where P_r is calculated by the same algorithm as P, but using trajectory parameters and error growth rates pertaining to the specific resolution trajectory.

The method of resolution adopted here consists of specifying a post resolution conflict probability, P_{rs}, and then synthesizing a resolution trajectory that achieves it with the smallest value of t_{rc}. It can be shown that values for P_{rs} in the range of 0.05-0.15 yield effective resolution trajectories.

Because of the non-linear, multi-parameter dependence of P_r on the resolution trajectory, it is generally not possible to obtain a closed form solution for a resolution trajectory as a function of P_{rs}. However, solutions can be generated easily by first parameterizing the resolution trajectories using one or two parameters and then iterating on the parameters until the conditions for an acceptable solution are achieved. In a 2D resolution, for example, suitable parameters are the change in heading

angle relative to the current heading, and the time at which to resume flight in the direction of a waypoint along the original route.

The procedure will be illustrated for a predicted collision ($d_{min} = 0$) at a 90 deg crossing angle, with both aircraft flying at 480 knots. Figure 9 shows the geometry of the conflict trajectories as well as the geometry of a horizontal resolution trajectory that minimizes both the deviation from the original route and the increment in flight time. This type of resolution trajectory can be used for route crossing angles between about 30 deg and 140 deg. It is characterized by two parameters, the time-to-go, t_{rc}, to the point of minimum separation, and the required minimum separation distance, $d_{r,min}$. Under the assumption that the prediction error growth rate for the resolution trajectory is the same as for the original conflict trajectory, and that the difference in the geometry of the trajectories is small, as is the case here, the post-resolution conflict probability function can be adequately approximated by the pre-resolution conflict probability function. These conditions allow one to calculate the required minimum resolution distance, $d_{r,min}$, as function of time-to-go, t_c, to the point of predicted minimum separation, d_{min}, for any value of P_{rs}, using a single family of curves such as shown in figure 5. Thus, if the post-resolution conflict probability, P_{rs}, is chosen to be a particular value, then $d_{r,min}$ can be found for any t_c by interpolation at P_{rs} on the family of curves in figure 5. This process can easily be automated for commonly occuring conflict situations. The middle curve in figure 10 represents a solution obtained by this process for $P_{rs} = 0.1$ and an along-track error growth rate of 15 kts. The solutions for 10 and 20 kts are also shown.

Note that the greater the error rate is, the steeper is the rise of the $d_{r,min}$ curves. Equivalently, for a fixed value of prediction time, the lower the error rate is, the smaller and less costly the resolution maneuver can be to achieve a specified value of post-resolution conflict probability. This result clearly shows the operational and economic value of increased prediction accuracy in the design of a conflict resolution system.

The useful range in prediction time to minimum separation of each solution is limited to a maximum value, $t_{r,max}$, determined by the specified value of false alert probability, P_{fa}. The effect of two values of false alert probabilities, $P_{fa} = 0.2$ and 0.3, on $t_{r,max}$ are investigated in figure 10. For the 15-kt error growth rate, the value of $t_{r,max}$ corresponding to a chosen value of P_{fa} is obtained from figure 5 as follows. Locate the probability value of 0.8 or 0.7, corresponding to a false alert rate of 0.2 or 0.3 on the ordinate and determine the point of interception of these ordinate values with the appropriate d_{min} curve. Then read off the corresponding values of $t_{r,max} = 15$ min and 18.7 min on the abscissa.

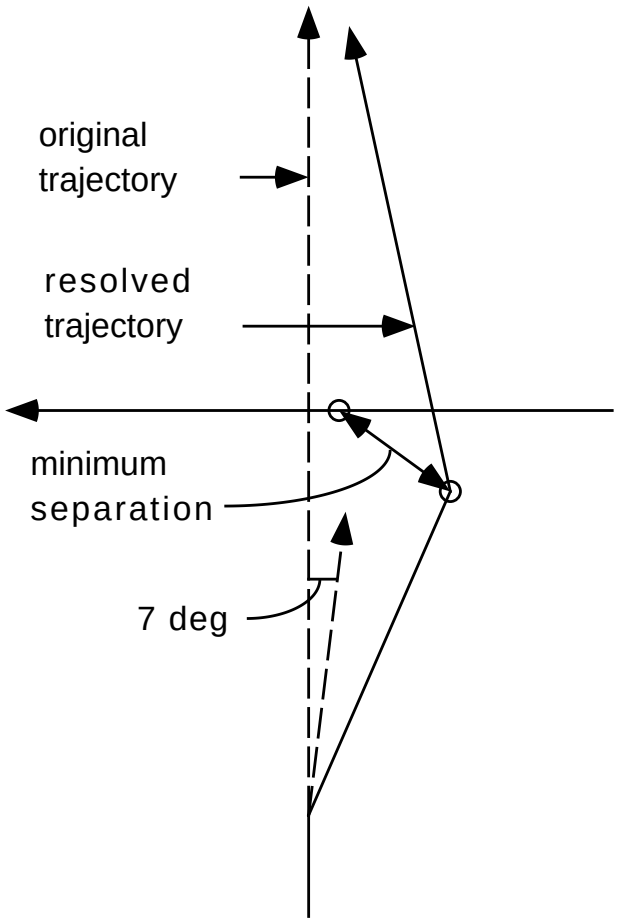

Figure 9: Conflict resolution example

These values of $t_{r,max}$ and the values corresponding to the other two error growth rates can now be entered on the abscissa of figure 10 to determine the corresponding minimum values of $d_{r,min}$. When the three loci of points on the $d_{r,min}$ curves are interconnected, a nearly straight horizontal line is obtained for each value of false alert rate, as shown in figure 10. This result simplifies the logic of the resolution process. The maximum value of $d_{r,min}$ depends on P_{fa} but is independent of error growth rate and $t_{r,max}$. For example, for $P_{fa} = 0.2$, the maximum value of $d_{r,min}$ is a constant 10 nmi while $t_{r,max}$ changes from 11.3 min to 22.5 min as the error growth rate decreases from 20 kts to 10 kts.

Finally, it is of interest to note that the maximum value of $d_{r,min}$ of 10 nmi is twice the value of the legal separation of 5 nmi. The 5 nmi buffer protects against errors in the execution of the resolution trajectory, which includes the usual trajectory prediction errors.

The resolution trajectory for this case is shown in figure 9. It is drawn to an exaggerated scale in the cross

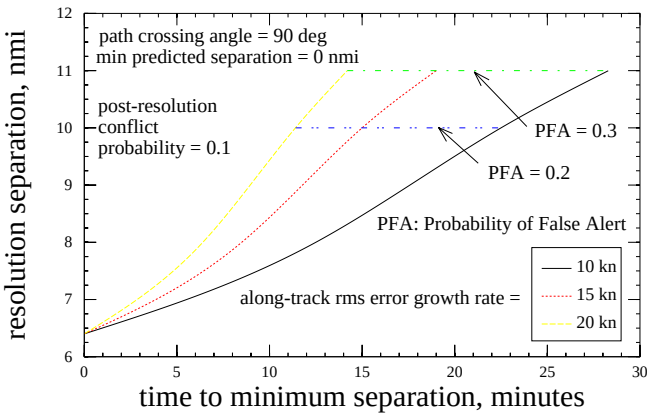

Figure 10: Resolution separation vs. prediction time

track direction in order to reveal the geometry more clearly. The actual deviation angle relative to the original route is only about 7 deg and is shown to scale in order to indicate how little the resolution maneuver deviates from the original geometry in this case. It can be shown that a resolution trajectory of the type shown minimizes the additional distance to fly and that the additional distance approaches zero as the resolution is initiated farther ahead of the conflict.

Finally, the functional dependence of $d_{r,min}$ on t_{rc} is seen to be slightly non-linear. However, in a real-time algorithm for automated conflict resolution, a linear approximation of this relationship would be entirely justified, thereby yielding a closed form solution for $d_{r,min}$ as a function of t_{rc}.

An alternative resolution method based on an optimal control formulation of this problem has also been developed [6]. It involves formulating a cost function for evaluating the cost of resolution maneuvers and then calculating the manuever that minimizes the cost. In addition to generating "optimum" resolution manuevers, the method also has the advantage of yielding the false alert and post resolution conflict probabilities, P_{fa} and P_{rs}, respectively, as part of its solution. When this method was applied to the example problem described earlier in this section, it yielded values of approximately 0.2 and 0.1, respectively, for these parameters. These results were the basis for the choices of P_{fa} and P_{rs} in the example problem. Furthermore, the functional dependence between $d_{r,min}$ and prediction time for the optimum resolutions were also found to be nearly linear. This result further justifies the linear approximation proposed in the preceding method.

Architecture of Conflict Probe

A Conflict Probe, including the conflict probability algorithm, has been implemented as an automation tool within CTAS. It shares approximately 90% of the software with the other CTAS tools (TMA, DA, FAST).

The foundation for all CTAS tools, including the Conflict Probe, is the four-dimensional (4D) trajectory modeling, analysis and computation module referred to as trajectory synthesis (TS) and route analysis (RA). The TS/RA algorithms generate predicted 4D trajectories for every aircraft within a region of airspace, which may consist of an enroute Center, a terminal area such as a TRACON, or a combination of enroute and terminal-area airspace. In order to serve the different requirements of each CTAS tool with a single, unified trajectory analysis system, the TS/RA design uses a scalable multi-workstation architecture to achieve both high computational speed and high prediction accuracy. TS/RA recomputes (refreshes) the predicted trajectories of all radar-tracked aircraft with an update cycle that essentially matches the radar update cycle of 12 s. For Center tools that use tracking data from enroute radars, the trajectory update cycle is approximately 10 s. It is about 5 s for terminal area tools using terminal area radars. The rapid update of trajectories ensures that all available sensor information is utilized in the calculation of the predicted trajectories. Therefore, CTAS advisories, including the conflict list derived from these trajectories, will adapt to changes in the current states of aircraft as rapidly as these state changes are reflected in the radar track data, flight-plan amendments, or other parameters. The rapid update cycle of 4D trajectories has proven to be fundamental to the operational effectiveness and controller acceptance of all CTAS tools. It has also been found to provide specific benefits for the Conflict Probe.

A block diagram of the CTAS architecture is shown in figure 11. The diagram is a simplification of the actual software architecture in that it does not explicitly show the communication-oriented processes or their interaction with the algorithmic processes. Each of the blocks in the diagram represent separate Unix processes which may be run on separate workstations or processors. In the NASA prototype, the system is implemented on a network of Sun Ultra workstations. The number of processors dedicated to TS/RA can be scaled to the maximum number of aircraft tracks that must be handled. For a single enroute Center with a maximum number of tracked aircraft of between 300-400, the four-processor configuration shown in figure 11 is adequate to update all 4D trajectories within 6 s of receiving each radar track. Additional processors can be added if the track count exceeds 400. The TS/RA processors generate trajectories asynchronously, thus emulating the operation of

a parallel computer. Load-balancing software automatically maintains an equitable distribution of tracks among the available processor. A TS/RA processor can also be added or removed without disrupting real-time operation. The TS/RA algorithm running in each processor contains about 110k lines of C code.

The 4D trajectories generated by the TS/RA processor are accessed by the software modules that generate controller advisories for the three CTAS tools, UPR/DA, TMA and FAST. Each tool has associated with it the special algorithms that generate advisories for that tool. Thus, FAST advisories are generated by an algorithm that optimizes runway allocation and arrival sequences. FAST also has its own embedded conflict detection and resolution scheme. TMA advisories are generated by a dynamic scheduling algorithm that meters traffic through meter gates. Finally, UPR/DA's advisories of predicted conflicts and trial resolutions are generated by a conflict detection and analysis module that will be described further in the next section. The software also provides graphical user interfaces (GUI) for each tool. The acronyms for the interfaces and the controller specialists who use them are shown in the figure. For FAST and TMA the interfaces are used by the traffic management coordinators. For UPR/DA the interface provides the capability that would be required by an area supervisor or sector controller. The code count in thousands of lines for each tool and process is also given in figure 11. The communications process (not shown in the figure) adds another 100K lines to the code count.

While all three tools collectively depend on the trajectory data generated by the TS/RA processors, the tool-specific processors are independent of each other and run asynchronously on separate workstations. The modularity of the CTAS hardware and software architecture offers several installation options, which can be used to trade off reliability on the one hand and economy and integrability on the other. For example, if UPR/DA and TMA are installed at a Center on independent networks of workstations, including separate TS/RA processors, software or hardware failures in one network will impact the operation of only the tool served by that network. The increased operational reliability of this configuration, however, has the disadvantage of making the integration of TMA and UPR/DA functions, which is required for the operation of the Descent Advisor (DA), far more difficult. Alternatively, if UPR and TMA are installed on the same network, as shown in figure 11, the number of workstations required to operate the tools is reduced significantly. The most important advantage of this single network configuration is that it greatly simplifies the integration of conflict probing and traffic management (TMA) as required for the operation of the DA tool. The main disadvantage is that failure of the network or software would cause both tools to fail.

Trajectory Modeling

Since knowledge of 4D trajectories is a basic requirement for all automation tools in CTAS, much effort has been devoted to developing computationally efficient and accurate methods for synthesizing such trajectories. The problem of synthesizing 4D trajectories and evaluating their prediction accuracy has been examined in a series of reports and papers [7, 8, 9]. The real-time algorithm based on these methods has been a part of the CTAS software since 1990 and currently supports the operation of CTAS at five enroute Centers and five TRACON's. Its prediction accuracy is under continuous scrutiny at NASA and the field sites, and performance enhancements are incorporated in periodic updates of CTAS software. Its accuracy for conflict prediction was evaluated in flight tests at the Denver Center in 1996 [10]. Here a brief review is given of the analytical and computational methods for generating the trajectories and of the considerations that led to the choice of these methods.

The trajectory synthesis in CTAS uses point mass equations of motion to model vertical and longitudinal accelerative maneuvers and concatenated segments of straight lines and circular arcs to model horizontal maneuvers and flight paths. The equations of motion are

$$\dot{V}_T = (T - D)/m - g\sin\gamma_a - \dot{V}_w\cos\gamma_a \quad (6)$$

$$\dot{h} = V_T\sin\gamma_a \quad (7)$$

$$\dot{S} = V_g \quad (8)$$

$$L = mg \quad (9)$$

$$R = V_g^2/g\tan\phi_{max} \quad (10)$$

where V_T is true airspeed, T is thrust, D is drag, L is lift, m is mass g is gravitational acceleratiopn, γ_a is flight path angle relative to airmass, V_w is wind speed component in flight direction, h is altitude, S is distance along flight path, ϕ_{max} is the maximum bank angle, R is the radius of a circular turn, and V_g is ground speed.

The force balance equations 6 and 9 directly relate the normal/vertical and longitudinal accelerations to the principal forces on an aircraft, namely, lift, drag and thrust. Furthermore, the pseudo-force contributed by altitude-dependent horizontal wind fields when an aircraft is changing altitude is accounted for by the last term in equation 6. This wind shear force plays a significant role in the accurate prediction of climbs and descents.

Horizontal paths are synthesized by a specially developed "circle geometry engine," which consists of a set

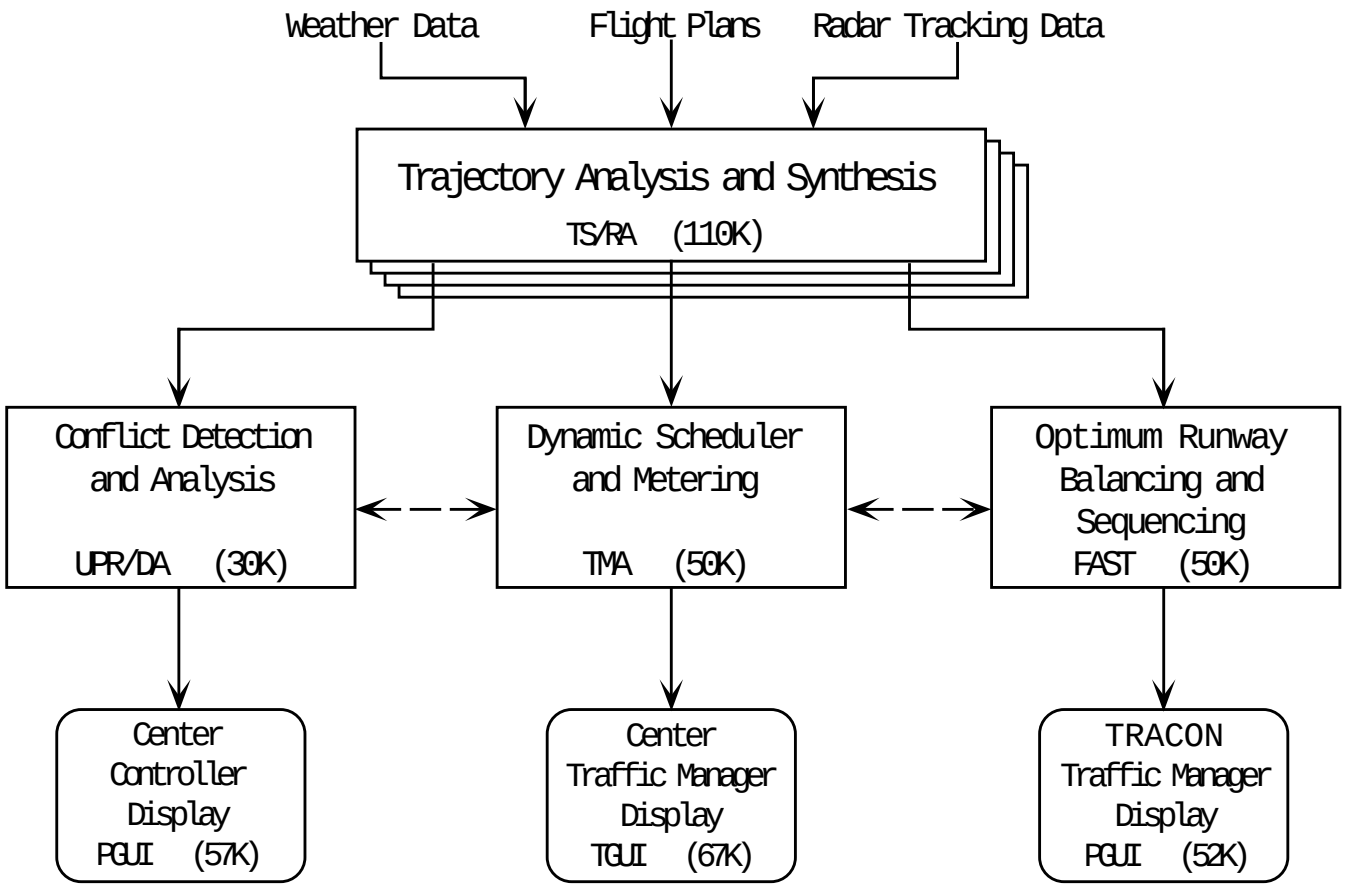

Figure 11: Center-TRACON Automation System (CTAS) Architecture

of closed-form (non-iterative) algebraic and trigonometric functions [8]. The engine generates horizontal paths of concatenated lines and tangent circular arcs as solutions to various types of area navigation and traffic management problems. These include waypoint-to-waypoint guidance, path stretching maneuvers, and many types of curved approaches and climbouts. Radii of circular turn arcs are calculated from the maximum allowed bank angle and an estimate of the ground speed in the turn, as given by eq. 10.

Trajectories are generated by numerically integrating the three differential equations using a 2nd order Runge-Kutta method with variable time-steps [7]. For generating climbs and descents, thrust levels and functions relating calibrated airspeed/Mach number to altitude must be specified. Flight path angle may also be used as a climb/descent control parameter. Ground speed is calculated from true airspeed, wind speed, wind direction, and airspeed direction by solving the well known wind triangle equation [8]. The position in the horizontal plane is determined from the known value of S at any time and the parameters defining the horizontal flight path.

In order to generate all types of trajectories for hundreds of aircraft at a high update rate, the trajectory synthesis software has been written to achieve both high computational speed and high accuracy. For example, if the synthesis algorithm is generating a trajectory segment wherein the airspeed, altitude and heading are being held constant such as in cruise flight, it skips the aircraft performance models and increases the integration step size to 240 s. If the trajectory segment is a deceleration, it uses the performance model and reduces the step size to 30 s.

Since the model equations include the lift, drag and thrust characteristics that are unique to each aircraft, synthesized trajectories based on these equations are guaranteed to stay within an aircraft's flight and performance envelope, resulting in flyable trajectories. This characteristic of the model has contributed perhaps more than any other to the operational effectivenes of the CTAS automation tools. It ensures that air traffic

clearances, pilot procedures, and airline preferred operational procedures during climbout and descent are transformed into 4D trajectories that are both flyable as well as likely to flown by a particular aircraft. Furthermore, trajectory synthesis that incorporates models of aircraft performance and forces emulates trajectory algorithms in modern flight management systems (FMS), thus permitting CTAS to accurately replicate FMS guidance modes. Recent field tests have demonstrated that the trajectory synthesizer in CTAS can be configured to accurately predict the climb and descent trajectories of aircraft both equipped and unequipped with an FMS. [9]. In summary, extensive operational experience has shown that the method gives accurate trajectories over the full range of problems encountered in normal aircraft operations.

Conflict Search Algorithm

An efficient conflict prediction and analysis algorithm for the UPR/DA Conflict Probe has been developed and implemented within the CTAS architecture and software. The design of the algorithm was driven by 3 criteria: efficiency, user flexibility, and completeness. These criteria and their influence will now be discussed.

Efficiency is the most influential factor in the design of the conflict prediction algorithm. The search for conflicts requires that every aircraft in the enroute Center be probed for conflicts against every other in less than the radar update cycle of approximately 12 s. Furthermore, enough time must be left in the radar update cycle for processing of inter-process messages, advisories, and trajectory updates. 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 translates to nearly 80,000 pairwise comparisons for 400 aircraft. 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. It has been determined that solving explicitly for minimum separation is not feasible for complex 4D trajectories and high traffic volume. 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 UPR/DA employs discrete comparisons, but with measures taken to limit the number of calculations required. Three means of limiting computational load are employed in the search algorithm: trajectory pair pruning, minimizing separation computations, and time-skipping.

Trajectory pair pruning determines if two trajectories are potentially in conflict and in need of further examination. This is accomplished through a simple set of heuristics which determine if the trajectories are spatially exclusive in either altitude or horizontal position. For example, if two enroute aircraft are cruising at assigned altitudes separated by at least the minimum required altitude separation, and will continue to fly at those altitudes throughout the airspace, there is no potential for conflict, and no need for discrete separation comparisons along the trajectories. Similarly, if two horizontal trajectories are enclosed in rectangles, and these rectangles are horizontally separated by the required amount, there is no potential for conflict. Every pair searched is first tested for altitude separation; if a con1flict cannot be ruled out by the altitude test, the horizontal rectangle test is performed next. The horizontal rectangle test is comprised of testing for y (northward) separation first, followed by a test for x (eastward) separation (due to the majority of traffic flowing East-West). The fraction of trajectory pairs which are pruned with these two tests varies with traffic mixture (ascents, descents, cruise), but is approximately 60-80% of all trajectory pairs. Thus, for 400 aircraft, the set of trajectory pairs remaining to be tested further is between 16,000 and 32,000 pairs. While this number is considerably reduced, the remaining pairs must be tested by comparing 4D trajectories, a computationally costly process.

Synthesis of 4D trajectories by the TS/RA process yields state information for each aircraft at time steps of 10 s along the trajectory. The components of the state vector used in the conflict search are: x, y, h, and time. The first method used to reduce computational load in the 4D search is to minimize separation computations. This is a process of eliminating computations based on the result of computations already performed for a time step. For instance, if at a given time step, two aircraft are separated by more than the required altitude, it is not necessary to calculate the horizontal separation. Furthermore, the sum of the squares of x and y separation is only calculated if altitude separation is less than required, and if both x and y separation is less than the required horizontal separation; the sum of the squares is only compared to the square of the required separation if it is the minimum separation thus far on the trajectory. By this process, it is only required to compute the minimum horizontal separation once for each pair in conflict.

In the third process of reducing computational workload of the 4D search, referred to as time-skipping, steps (described above) are skipped if it is determined from the previous step that no conflict is possible given

aircraft performance. For example, two aircraft separated horizontally by 200 nmi, at a given step, will not be in conflict on the next step. Conservative estimates of aircraft performance are used to determine how many steps can be skipped. These estimates are in the form of assumed maximum altitude and horizontal closure rates. Altitude closure is assumed not to exceed 10,000 ft/min, while horizontal closure rate is assumed to be less than 0.33 nmi/s, or about Mach 2 at standard sea level conditions. These closure rates are used to calculate the increment in time at which there is potential for a conflict. The greatest increment in time to potential conflict, determined from the horizontal and altitude closure rates, is converted into an equivalent number of steps to skip, and the search continues at that time on the trajectories. This process is handled efficiently by storing the time steps in a dynamically allocated array of pointers rather than a linked list, allowing direct access to time steps via the array index (time step index).

The design requirement for adjustability of minimum required altitude separation as a function of flight phase (level flight, non-level flight) meant required separation for a trajectory pair could change during the search of a trajectory pair. This is accounted for by creating a list of separation transition times and separation values corresponding to each change in required altitude separation throughout the duration of the search, prior to pairwise comparison. A single time comparison is needed to determine the required separation, rather than evaluation of the flight phase of each trajectory, at each time step.

Finally, conflict prediction software must be complete and dependable. While some missed conflicts are unavoidable (e.g. when the intended path of the aircraft is unknown or is changed after a conflict search), it is essential to take measures which guarantee that all potential conflicts are detected in every search cycle. The simplicity of the search algorithm, and the conservative choice of parameters used in the pruning process ensure completeness and dependability.

Once a potential conflict pair has been identified, the last step in the detection process consists of computing the conflict probability for the pair. The conflict probability computed during each conflict search cycle (6 s) is based on the most recent 4D trajectory predictions and thus reflects changes in predicted minimum separation due to radar noise, etc. The raw conflict probability values are smoothed by a first order lag filter prior to evaluation by the probability filter decision logic described in a preceding section. A filter time constant of 18 s has been found to provide an acceptable compromise between responsiveness and smoothness. Furthermore, the software contains logic to enhance the temporal stability of the conflict list by restricting the list to include only those predicted conflicts which continue for a specified number of consecutive search cycles.

Two types of conflict searches are implemented using the preceding algorithm: global search and trial planning search. The global search employs a 12 s update rate and searches the $n(n-1)/2$ set of trajectory pairs previously described. The trial plan search operates on a rapid, 1 s update due to response time requirements. Trial planning is a function used by controllers to evaluate if a controller's proposed resolution of a conflict is itself conflict free. A trial plan consists of a controller-selected change in speed, altitude, and/or route of an aircraft currently in conflict. An aircraft selected by a controller for trial planning is handled by a separate search procedure. This procedure requires only a 1-vs-n aircraft search but a much higher update rate of 1 s. The high repetition rate of the trial planning search gives nearly instantaneous response to controller changes in the trial plan.

The conflict prediction process has been implemented in a CTAS software process called PFS_C, containing approximately 20K lines of C code, including the conflict search process (approximately 3K lines). It constitutes the largest element in the UPR/DA block of figure 11. PFS_C consists of the two conflict search procedures previously described (global and trial planning), the conflict probability algorithm (described in a previous section), and other functions handling advisory processing, inter-process messaging, and trajectory updates.

The conflict search algorithm has demonstrated the performance required for the Conflict Probe. The performance of the search algorithm is gauged by the time per trajectory pair, and has been measured to be in the 45-55 μs range (30 min prediction interval) for a Sun Microsystems Ultra 170E (340 MIPS). Approximately 40% of this time is spent performing pairwise 4D trajectory comparisons, while 10% is spent pruning trajectory pairs from the process. The remaining 50% of the time is accounted for by overhead: copying time steps into dynamically allocated arrays from linked lists and binary tree traversal to access aircraft information. This time per trajectory pair and the percentage of trajectories pruned from the prediction process can be used to determine the maximum number of aircraft that can be probed for conflicts within 6 s. This number is in excess of 800 aircraft and suggests that the Conflict Probe can handle the combined aircraft in two enroute Centers in less time than the radar update period of 12 s. Actual observed performance of the Conflict Probe receiving live tracking data from Denver Center has been 1.4 s to process 380 aircraft.

Concluding Remarks

A rational foundation for the design of a Conflict Probe with long detection time horizons has been developed by combining 4D trajectory prediction with analytical models of prediction errors and conflict probability. The CTAS software provides an efficient platform for the realization of this design. By utilizing 90% of existing CTAS software, the Conflict Probe required the development of only two new processes, a generic conflict search engine and the conflict probability algorithm, which together added about 30,000 new lines of C code to CTAS. Significant benefits arise from the insertion of a Conflict Probe within CTAS. It offers the flexibility of using the Conflict Probe as a standalone tool for Center controllers or using it as an integrated tool with the Decent Advisor and the Traffic management Advisor. Since the CTAS trajectory synthesizer generates accurate 4D trajectories for an entire flight profile, from liftoff to final approach, the Conflict Probe can be used in all regions of airspace. This helps to remove significant boundary constraints between enroute and terminal-area airspace. By allocating conflict search and 4D trajectory synthesis, the two most computationally intensive processes in conflict probing, to separate workstations, the Conflict Probe is able to search up to 800 aircraft tracks for conflicts within 10 s. Thus, the Conflict Probe benefits from the mature trajectory synthesis and scalable multiworkstation architecture of CTAS, which have been in operational use at several Centers and TRACONS for several years.

The Conflict Probe is fully merged within CTAS and will be included in future CTAS software releases starting in July 1997. It has been adapted to the airspace of the Denver and Fort Worth Centers, and field evaluation is scheduled to begin at both Centers this summer.

References

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

[4] 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.

[5] Ballin, M.G.; Erzberger, H.: "An Analysis of Landing Rates and Separations at Dallas/Ft. Worth Airport", NASA Technical Memorandum TM-110397, July 1996.

[6] Erzberger, H.: Air Transportation Systems Technical Committee Workshop; AFM In the 21st Century: Where Does Free Flight Fit In?, Washington D.C., March 6, 1996.

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

[8] 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.

[9] 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.

[10] McNally, B.D.: "Technologies for User Preferred Routing," 15th Digital Avionics Systems Conference, Atlanta GA, Oct 27-31, 1996.