A Self-Organizational Approach for Resolving Air Traffic Conflicts

Martin S. Eby

■ **The use of airways and navigational fixes to form fixed routes in the sky is central to modern-day air traffic control (ATC). Fixed routes, however, limit efficiency and airspace capacity in comparison to paths that are not constrained or predefined. This article describes a computational technique for determining collision-free time/space paths for multiple vehicles without the use of fixed routes. The technique applies a simple destination-seeking rule and a few conflict-avoidance algorithms to each vehicle individually such that the collective solution is determined by the calculated behavior of the individual vehicles; that is, the overall solutions are self-organizational in nature. This self-organizational approach has been tested in a variety of scenarios ranging from simple two-dimensional conflicts to the modeling of an ATC sector handling an unrealistically high traffic load. The simulations, implemented on a very modest computer workstation, have proven the self-organizational approach capable of finding solutions to complex traffic conflicts at rates faster than real time. Furthermore, the self-organizational solutions tend to require smaller speed/direction deviations than solutions obtained with human reasoning.**

A RECENT AIRMAN'S map is shown in Figure 1. The most prominent markings on the map are a number of radio beacons (shown as dark circles) interconnected by a number of air routes (shown as dark lines). The beacon/airways system arose from the limited technology that was available in the early days of instrument-based navigation; pilots could make their way cross-country without visual landmarks by using a simple single-channel receiver to fly along one radial toward a beacon, over the beacon, and then outward again along a different radial until reaching the radio reception range of the next beacon. With this technique, known as angle/angle navigation, the shortest possible routing occurs when the angle of exit from one beacon coincides with the angle of entry to the next beacon, i.e., moving from beacon to beacon to beacon in a straight line.

Today, aircraft are no longer limited to angle/angle navigation because of four-dimensional (three spatial dimensions and time) area navigation capabilities based on radio triangulation, the Long-Range Navigation (LORAN) system, or the Global Positioning System (GPS), among other systems. Such navigation capabilities have enabled the use of arbitrary routes, which offer a number of advantages over airways routing:

1. greater fuel efficiency,
2. reduced flight time,
3. increased airspace capacity, and
4. reduced potential for pilot error because the arbitrary routings have fewer waypoints than the equivalent airways-based routings.

But, despite the various advantages of direct/arbitrary routing and the fact that nearly all commercial (and many private) aircraft are capable of flying such routes, the vast majority of flights continue to follow airways. The reason lies not with the aircraft or the pilots, but with air traffic controllers, who rely on the airways structure for maintaining safe spacings between aircraft.

Airways simplify the controller's job by limiting the trajectories that aircraft may follow, thus also lim-

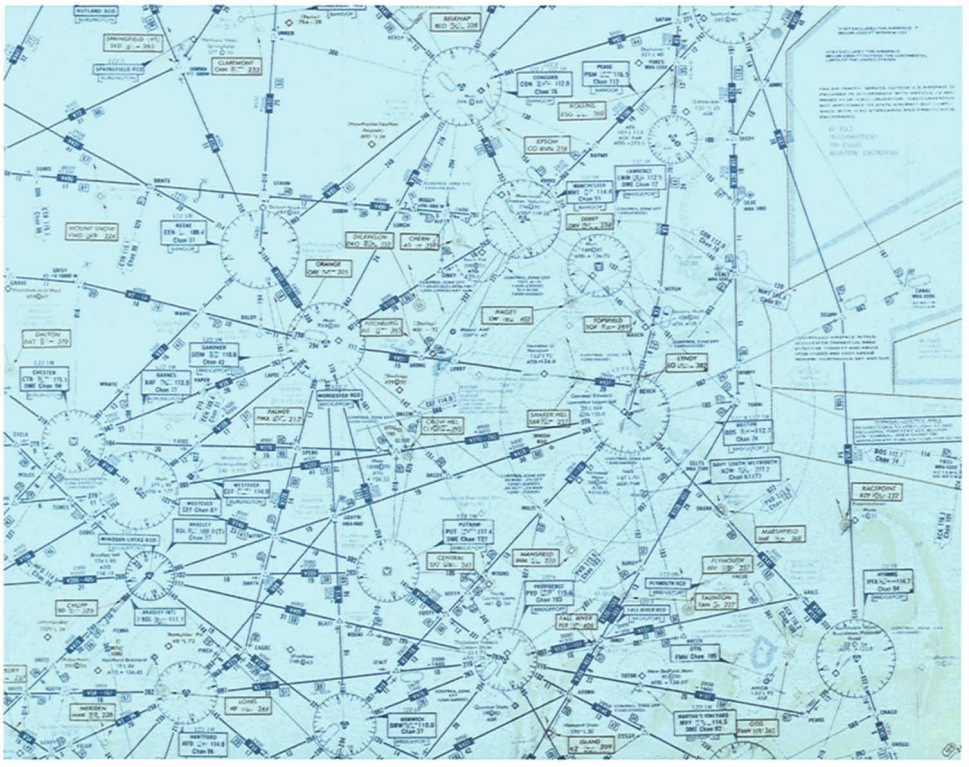


FIGURE 1. Airman's map showing radio beacons (dark blue circles) and air routes (dark blue lines connecting the dark blue circles).

iting the trajectories on which the aircraft may collide. This restriction essentially reduces the general three-dimensions-plus-time conflict-resolution problem to a series of one-dimension-plus-time problems. The latter type of problem is much easier for humans to solve, with the one drawback being that the possible solution set will be small relative to the three-dimensional problem. Indeed, without the use of airways, the current ATC system would simply be unable to handle typical daily traffic. (This is not to say that aircraft are never granted direct routing. In fact, controllers often use direct routing during the early morning when traffic densities are low enough that controllers may handle the mental calculations required for all-aspect conflict detection and resolution.)

For the reasons outlined earlier, the development of an air traffic routing paradigm that does not confine aircraft to the airways network promises fuel and time savings, greater safety, and reduced pilot work load. In the absence of a breakthrough technique by which humans can detect and solve the general-case three-dimensions-plus-time conflicts, we may assume that any such alternative to the current ATC system will involve the use of computers, the decentralization of control authority, or both. Various approaches may be taken in an attempt to find sufficiently optimal solutions to multi-aircraft conflict resolution in the absence of the airways system's constraints.

This article describes a self-organizational approach to aircraft conflict resolution in which the aircraft are modeled on an individual basis and each modeled pilot/aircraft is concerned chiefly with achieving a solution that satisfies the needs of that particular aircraft rather than working to achieve a solution that works for all aircraft. Although the implementation of this approach would cause a revolution in ATC techniques, the basic concept is most certainly not a revolutionary method of traffic conflict resolution—we, as individuals, engage in self-organizing traffic conflict resolution whenever we drive a car on a highway or walk through a crowded shopping mall.

This article does not propose to make air traffic *truly* self-organizing. Rather, it describes a system wherein *centrally managed* ATC may provide safe and efficient routings by implementing a computer-based

self-organizing model of the air traffic. Aircraft would still receive routing instructions from ground-based controllers; only the mechanism for determining the appropriate instructions would be changed. (To the extent to which the decisions of the modeled pilots agree with the decisions that their real-life counterparts would make given the same information, the behavior of the model will match the behavior of truly self-organizing air traffic. This feature makes the inclusion of pilot preference data or a migration toward truly distributed/self-organizing ATC straightforward relative to other approaches.)

Self-Organization and ATC

Loosely defined, a self-organizing system is one in which organization (e.g., multiple aircraft each reaching its destination without violating separation criteria) is achieved through the actions taken by the individual elements of the system rather than through a master plan imposed on the elements from outside the system.

The physical sciences provide at least one model of three-dimensions-plus-time traffic conflict resolution. In Figure 2, several positively charged particles have been released into a space that contains one fixed negative charge. The positive charges will tend to be drawn toward the fixed negative charge because of the mutual attraction of their opposite charges. At the same time, the positive particles tend to maintain distance between each other because of the mutual repulsion of their like charges. By treating the free-floating positive charges as aircraft and the fixed negative charge as an airport or waypoint (i.e., a destination), we have a crude model that is capable of moving aircraft toward their destination while avoiding collisions.

Pilot Algorithm

The potential-field model described above is much too simplistic for application to real-world ATC. For example, in the potential-field model the attractiveness of a destination varies with distance from the destination. Moreover, the separation maintained between particles is a function of their closing speed (with higher rates of closure leading to smaller separations), whereas in ATC scenarios the separation must be independent of closing speed; that is, there should always be a minimum separation maintained between aircraft regardless of the speed of the aircraft.

At Lincoln Laboratory, an algorithm (Figure 3) has been developed that retains the basic attraction/repulsion feature of the potential-field model but is useful for solving air traffic conflicts. The algorithm, which is applied to each aircraft simultaneously, consists of the following steps:

1. Determine the distance/direction that would be traveled in the absence of any traffic conflicts. Call this vector **D**. If we refer to the ideal speed as s_{ideal}, then the ideal path vector is $s_{ideal}\,\mathbf{D}/|\mathbf{D}|$.
2. Project the current time/position plans of the other aircraft to determine which aircraft will intrude into the airspace occupied by the subject aircraft in the future. Call these *obstacle aircraft.*
3. For each obstacle aircraft:
   a. Determine the function $i(t)$ that describes the amount (scalar) of intrusion, where intrusion is defined as the difference between the desired separation r_D and the projected separation $|\mathbf{R}(t)|$. Note that the desired separation need not be constant; it

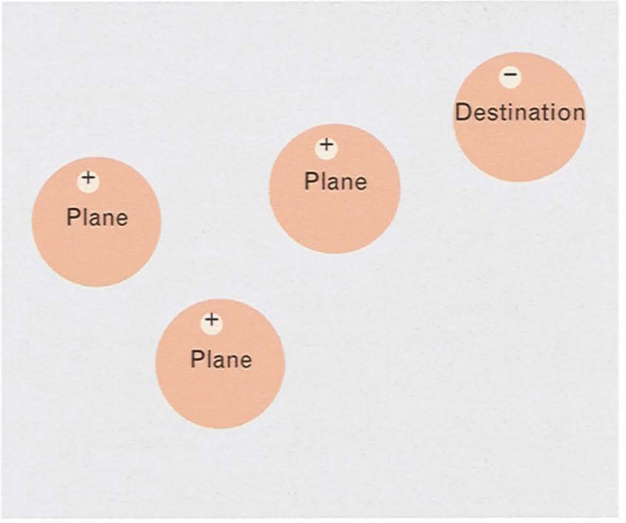


FIGURE 2. Air traffic model based on voltage potential fields. The model exhibits two critical features of successful ATC: aircraft (positively charged particles) tend to maintain separation between each other (because of mutual repulsion) while moving toward their destination (because of attraction to the opposite charge).

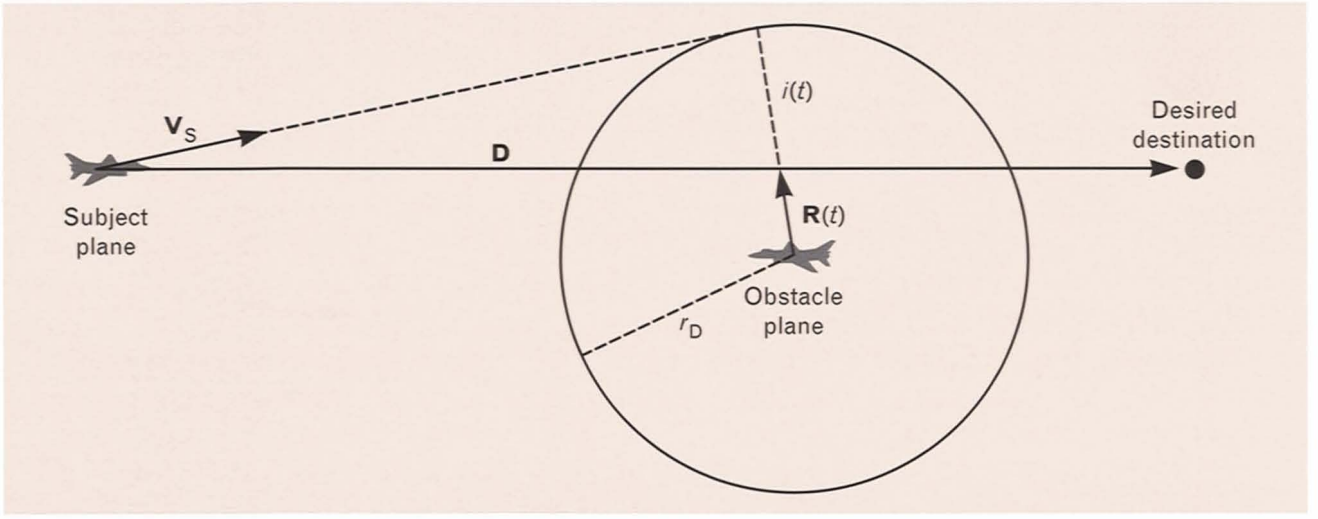


FIGURE 3. Geometry of algorithm for resolving air traffic conflicts, where **D** is the vector from the subject's current position to the desired destination, r_D is the desired miss distance, $\mathbf{R}(t)$ is the vector from the obstacle's future position to the subject's future position, $i(t)$ is a scalar quantity equal to the greater of 0 and $r_D - |\mathbf{R}(t)|$, $\mathbf{V}_S$ is the solution velocity vector, and t is time. For a detailed description of the algorithm and the mathematics involved, see the subsection entitled "Pilot Algorithm," which begins on the previous page.

can vary to accommodate uncertainty and other properties or effects.

b. Determine the time t^* when the value of $i(t)/(t - t_{now})$ is a maximum. In words, the quantity t^* can be thought of as the time when the size of the conflict is largest relative to the time remaining to solve the conflict.

c. The minimum spatial differential that eliminates intrusion at time t^* is then given by $i(t^*)\mathbf{R}(t^*)/|\mathbf{R}(t^*)|$ and the change in the subject plane's course/speed is given by $i(t^*)\mathbf{R}(t^*)/[|\mathbf{R}(t^*)|(t^* - t_{now})]$. We refer to this result as the *avoidance vector.*

4. Sum the calculated avoidance vectors and add to $s_{ideal}\mathbf{D}/|\mathbf{D}|$ to yield the solution vector $\mathbf{V}_S(t_{now})$. (Note: Summing the avoidance vectors to yield the solution vector is directly analogous to summing the repulsive forces in the potential-field model. The net effect is to maintain spacing between the aircraft, or positively charged particles.) Then, adjust $\mathbf{V}_S(t_{now})$ as necessary to satisfy the acceleration and velocity limitations of the subject aircraft.
5. Advance the model aircraft along $\mathbf{V}_S(t_{now})$ for a short time interval.

The above steps describe the algorithm for a single (modeled) aircraft for a single (modeled) point in time. The total solution requires that calculations be performed for each modeled aircraft at (modeled) time intervals that are small relative to the time remaining before a conflict involving the aircraft is projected to occur. In this way, the modeled pilots "see" and respond to the changes in various obstacle aircraft's paths as the solution is calculated. This process is similar to car drivers seeing and responding to traffic conflicts as they develop, e.g., traffic from an on-ramp merging with traffic on a highway. Using this algorithm, Figure 4 diagrams the resulting courses for two aircraft (one traveling at roughly twice the speed of the other) with paths that intersect at right angles and the constraint that neither aircraft may maneuver in altitude. Each red X represents a separate time step and calculation of an aircraft's path.

Features of the Basic Pilot Algorithm

The above algorithm is very simple, yet handles en route conflicts rather well and in an intuitively sensible fashion:

1. In the absence of traffic conflicts, the aircraft proceed directly to their destinations.
2. The response to a given conflict is appropriate

to the time proximity and magnitude of the conflict; that is, small conflicts far in the future result in very minor deviations in course and speed while larger and/or more immediate conflicts result in larger course/speed changes.

3. Each modeled pilot takes responsibility for the safety of his or her aircraft. In other words, each pilot reacts to a conflict as though the other pilot(s) involved in the same conflict would not. In Figure 4, for example, both of the aircraft paths initially display rather large angles of deviation from the ideal paths but, as each pilot "sees" the other acting to avoid the conflict, he or she correspondingly reduces his or her own deviation. (Note: Thus far, we have been attempting only to calculate conflict-free paths; optimization of the conflict-free paths will be performed in a later step.)
4. Multi-aircraft conflicts have also been handled with similar results. In multi-aircraft conflicts, aircraft that are only peripherally involved in a conflict are "repulsed" by one or more of the other aircraft in the conflict such that they pass even farther from the center of the conflict. By doing so, they permit more maneuvering room near the center of the conflict so that the aircraft involved there are spared the need to make extreme deviations. Figure 5 diagrams this behavior for the simple case of three aircraft that are not permitted to change altitude. Here, the algorithm's repulsion mechanism causes two southbound aircraft to increase the distance between each other so that a northbound aircraft may pass between while the net effect on the northbound aircraft is nil.

The two examples presented up to this point are quite simple. To be useful as an ATC aid, a self-organizational approach must also be able to solve a variety of less contrived problems that humans find difficult, even impossible, to solve in a timely fashion.

Algorithm Verification

We now consider a more complicated problem with eight aircraft traveling four different speeds from eight directions of the compass (Figure 6). As in the earlier examples, no maneuvering in altitude is al-

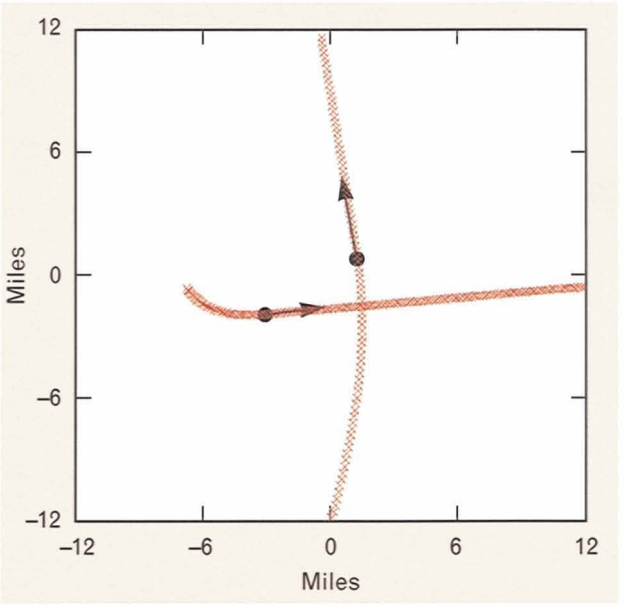


FIGURE 4. Conflict resolution for two-vehicle example using the algorithm of Figure 3. The red Xs indicate the positions of the aircraft at each modeled time interval, the black dots indicate the positions of the aircraft at the time of closest approach, and the black arrows indicate the relative speeds of the aircraft. Note that one aircraft is traveling at about twice the speed of the other vehicle. (In this example, altitude changes were not permitted.)

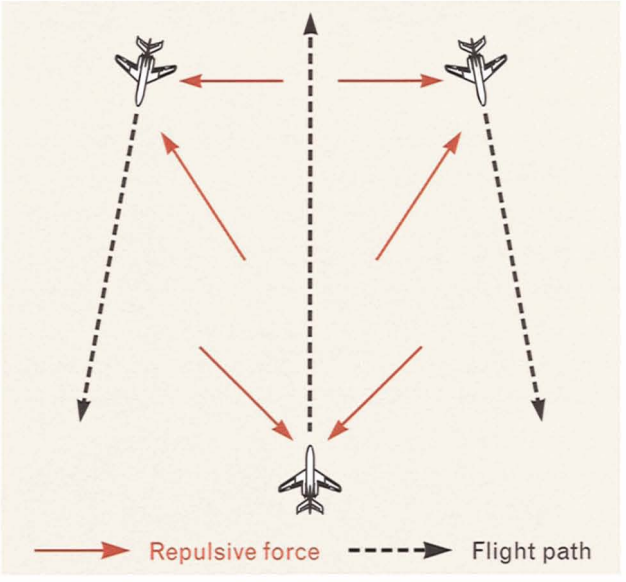


FIGURE 5. Multi-aircraft conflict resolution using the algorithm of Figure 3. Two southbound aircraft increase the distance between each other so that a northbound aircraft may pass between. Note that the net effect on the northbound aircraft is nil. (In this example, altitude changes were not permitted.)

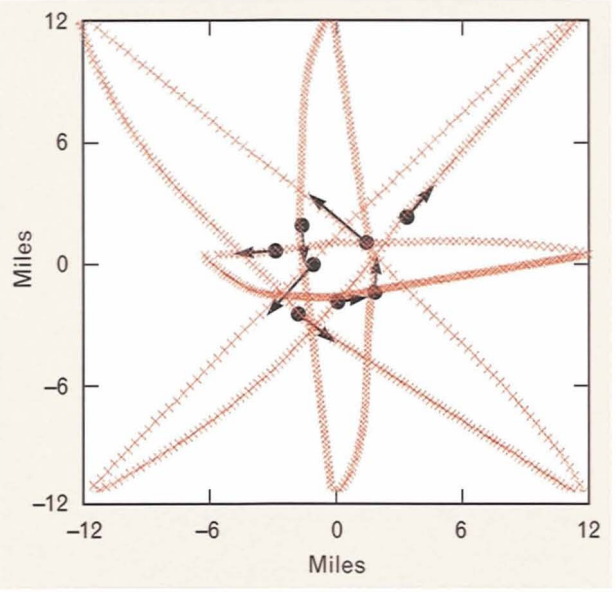


FIGURE 6. Conflict resolution for eight-vehicle example. The red Xs indicate the positions of the aircraft at each modeled time interval, the black dots indicate the positions of the aircraft at the time of closest approach, and the black arrows indicate the relative speeds of the aircraft. The example has been constructed such that, in the absence of any avoidance maneuvers, all the aircraft would arrive at the center of the plot at the same instant. Note that the solution requires comparatively little deviation on the part of any of the aircraft.

lowed; thus the problem is two-dimensional. Also, we have constructed the problem such that, in the absence of any avoidance maneuvers, all the aircraft would arrive at the center of the plot at the same instant. The arrows on the solution paths indicate the position and relative speed and direction of the aircraft at the moment when they would otherwise be colliding. Note that the solution requires comparatively little deviation on the part of any of the aircraft. Clearly, it would be difficult and very time consuming for a human controller to determine similarly efficient routings.

Random Conflicts

We now investigate the performance of the algorithm on randomly generated problems in which altitude may be used as a means of maintaining separation between aircraft. (Note: Altitude differences are scaled such that 1000 ft of vertical separation is equivalent to 5 mi of horizontal separation.) Fifty problems sets have been generated with the following steps:

1. Define a horizontal area in the sky—a square 24 mi × 24 mi—and assign a particular altitude to that area.
2. Randomly pick eight starting points at the chosen altitude inside the area such that the minimum distance between any two points is at least 5 mi.
3. Randomly pick eight endpoints at the same altitude such that the minimum distance between any two endpoints is at least 5 mi and each endpoint is at least 16 mi from its corresponding starting point.
4. At each starting point, place an aircraft that will travel at a nominal speed of between 80 and 240 kn. (This range of speeds is typical for aircraft in terminal control areas.)
5. Adjust the departure time of each aircraft according to the speed of the aircraft and the distance between the starting point and endpoint of that aircraft's path such that, in the absence of any conflict resolution, all of the aircraft will be at the midpoints of their ideal paths at the same time. (Note: This step has the effect of maximizing the congestion and difficulty of the problem.)

Figure 7 depicts a typical problem (before solution) that was produced by the steps described above. Again, this problem involves eight aircraft, all with originations and destinations inside a square area (24 mi × 24 mi) at the same altitude; that is, the congestion is extremely high. For this random problem, Figure 8 shows the solution obtained with the self-organizational algorithm. Note that, even for this extremely challenging scenario, the calculated paths do not exhibit particularly large detours; that is, the individual computer-modeled pilots do a good job of finding efficient solutions.

Figure 9 summarizes the conflicts of each of the 50 problems. The x-axis represents the 50 runs; the y-axis represents the closest approach between two aircraft in a given run. Because there are eight aircraft in each of the randomly generated scenarios, there are $7 + 6 + 5 + 4 + 3 + 2 + 1 = 28$ datapoints for each run. These datapoints are shown as Xs in the figure. Note that each of the scenarios involves numerous con-

flicts. (A conflict is defined as a situation in which two aircraft pass within 5 mi of each other.) Figure 10 provides a summary of the separation attained with the algorithm for the 50 scenarios of Figure 8. Note that the bottom of the graph is almost completely free of Xs, indicating the algorithm has achieved a dramatic improvement in aircraft separation.

Problems and Solutions

Although the overall conflict-resolution capability of the self-organizational algorithm is impressive, Figures 8 and 10 show three aspects that must be addressed before the algorithm can be used operationally:

1. Although the calculated paths diagrammed in Figure 8 are well within the flight capabilities of the aircraft being modeled, the paths are far too complex and detailed to be transmitted reliably to an aircraft and carried out by a pilot. (Note: A direct datalink to the flight management system would make such complex maneuvers feasible, but the optimization operations described in this section resolve this problem by reducing the complexity while increasing the efficiency.)
2. In Figure 10, the achieved separation is not cut off sharply at the targeted 5-mi distance. Indeed, many aircraft pairs have a closest-approach distance of just slightly under 5 mi.
3. Figure 10 also depicts a few outliers that had minimum separations well under 5 mi—only 80% or even 70% of the desired separation.

Simpler Paths

Although the paths that the model pilots calculate and "fly" can have many speed, altitude, and/or direction changes, the routes that the real-life aircraft are instructed to follow need not be as complex. If we consider the initial calculated path to be a trajectory that consists of many time/position waypoints, it becomes a simple matter to examine the path and look for waypoints that may be deleted without (a) causing the aircraft to enter into prohibited airspace, (b) creating a conflict with another aircraft, or (c) exceeding the performance limits of the aircraft. Figure 11 shows how the paths of the two aircraft of Figure 4 can be simplified to a single waypoint each. An im-

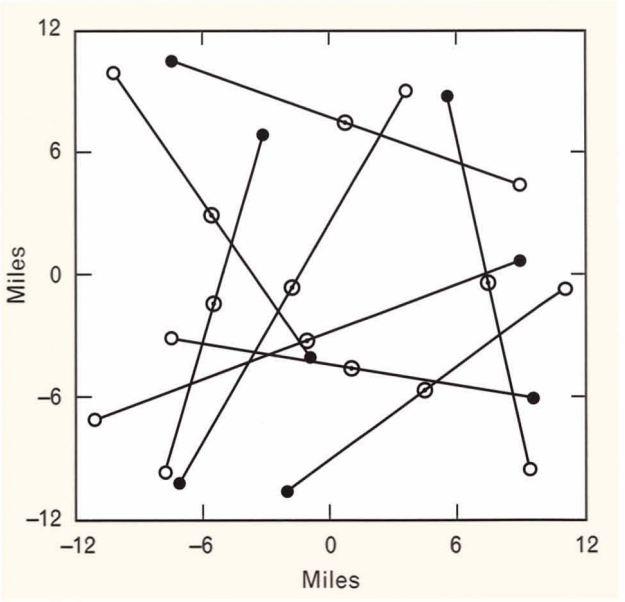


FIGURE 7. Random-conflict scenario for eight aircraft at the same altitude. The black dots represent the starting positions of the aircraft, the hollow black circles indicate the ending positions, and the black circles with dots inside of them denote the positions at the time of approximate maximum congestion in the absence of any correction.

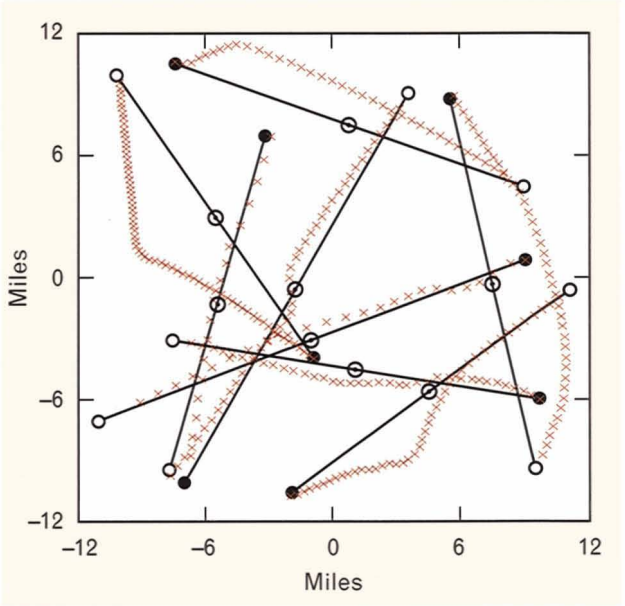


FIGURE 8. Solution to random-conflict scenario of Figure 7. A self-organizational algorithm was used to obtain this solution, with the red Xs indicating the paths taken by the aircraft at each modeled time interval. Note that, even for this extremely challenging scenario, the calculated paths do not exhibit particularly large detours.

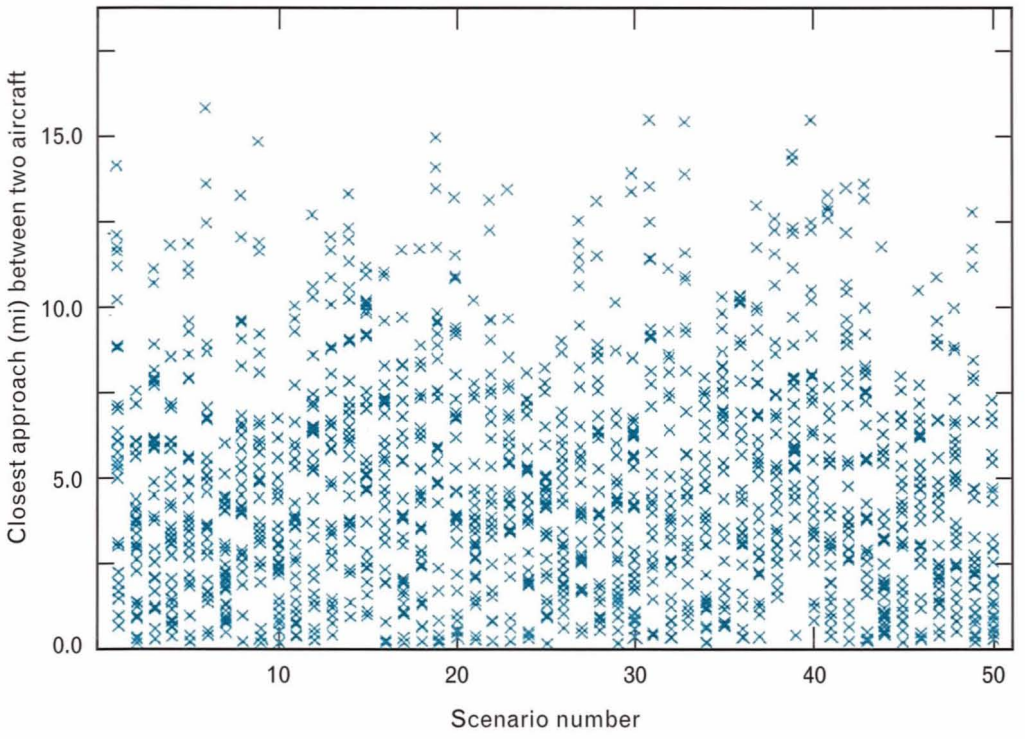


FIGURE 9. Summary of random conflicts without resolution for 50 scenarios, each involving eight aircraft (see Figure 7). Note that all of the scenarios involve aircraft flying within 5 mi of each other.

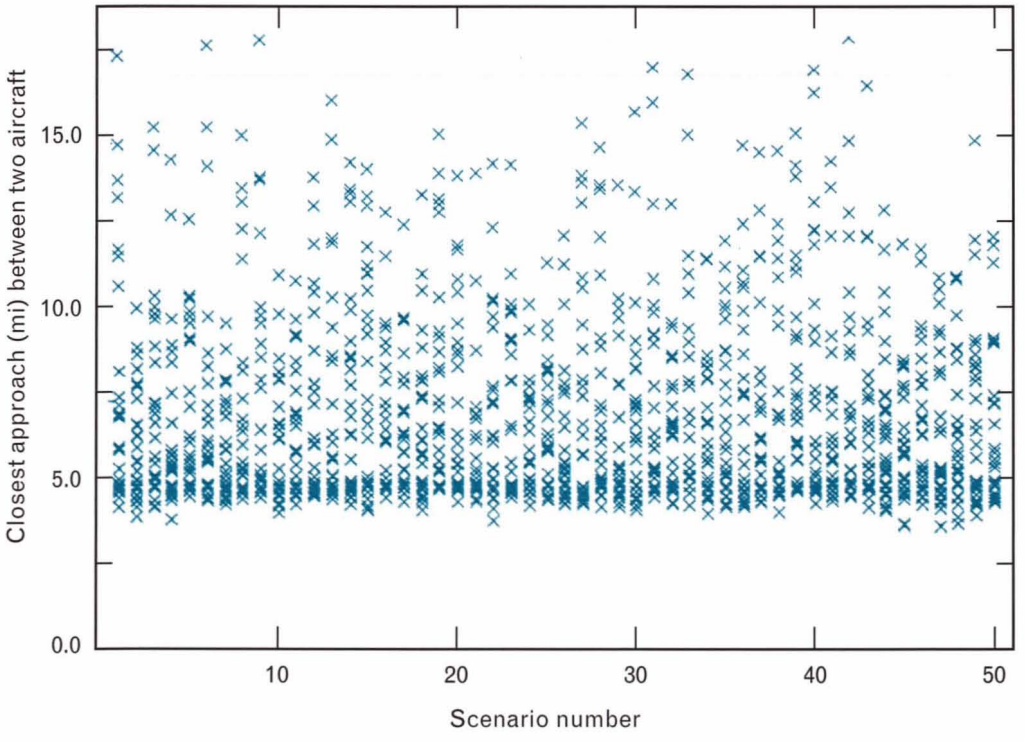


FIGURE 10. Summary of random conflicts with resolution for the 50 scenarios of Figure 8. A self-organizational algorithm was used to resolve the conflicts. Note that the bottom portion of the graph is now almost completely free of Xs, indicating the algorithm has achieved a dramatic improvement in aircraft separation.

portant and very useful secondary effect of this path simplification is a reduction in the overall length of the path that an aircraft must traverse, i.e., an increase in the efficiency of the solution. Also note that we can use this path simplification to eliminate as much or as little of the margin-of-error allowance as desired such that the allowance for the simulation artifacts described below may be removed at this step.

Simulation Artifacts

The algorithm employed by each modeled pilot seeks to resolve a conflict or conflicts by detouring an aircraft the minimum amount necessary from the ideal course and speed. In a scenario that involves maneuvering to avoid a fixed object, the pilot would achieve exactly the degree of separation that the algorithm aims for and no more. In a multi-aircraft conflict, however, the conflict itself evolves over time as the aircraft involved maneuver to avoid each other. The changing nature of the conflict combined with the sequential and discrete calculations of a computer simulation adds "noise" to the result such that the closest approach between two aircraft involved in a routing conflict will occupy a small range around the target separation value rather than matching that value exactly.

The magnitude of these errors, however, is small relative to other system errors and uncertainties in, for example, knowledge of winds aloft, the fidelity and timeliness with which a human pilot can respond to a controller's routing instructions, and even the starting positions of aircraft. (Note: In the 12-sec interval between sweeps by a long-range surveillance radar, an aircraft flying at a speed of 300 kn can travel a distance of a full mile.) In other words, the range of closest approaches that results from the discrete nature of the simulation is insignificant compared with the margin of error that must be added to the desired separation to compensate for errors and uncertainties elsewhere in the ATC system. Indeed, because the simulation eliminates human error/uncertainty, the size of the total error margin in the simulation may actually be reduced to the extent that the computer's calculations of a conflict-free path are more accurate than the corresponding calculations of a human controller.

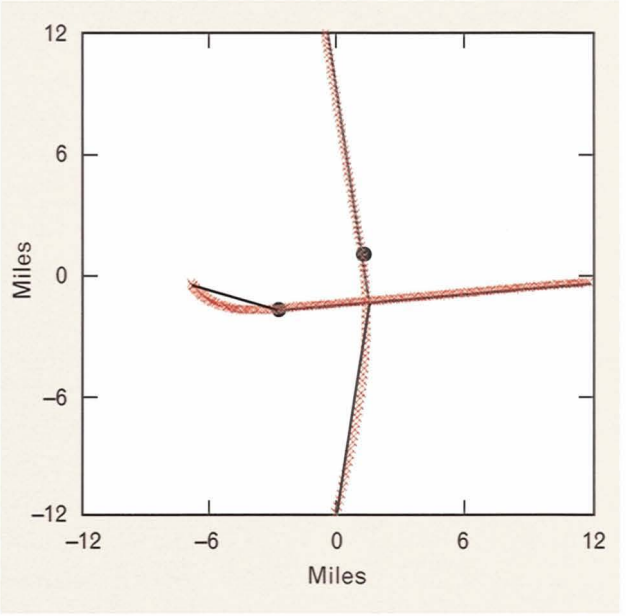


FIGURE 11. Path simplification of the solution given in Figure 4, with the black solid lines indicating the simplified paths.

Handling Failures

The third problem to be addressed is the case in which a flight path fails by a considerable margin to provide the desired separation. Figure 10 shows a handful of such cases, with the worst one resulting in a closest-approach distance of only 3.5 mi. Note, however, that even when the algorithm fails to find suitable paths for all the aircraft involved in potential conflicts, the failure is not catastrophic—the majority of aircraft do achieve the desired separation and the other aircraft at least maintain a degraded separation; that is, no direct collisions result.

The algorithm failures found in Figure 10 result from a combination of the following two factors:

1. If two or more aircraft act as obstacles on opposite sides of an aircraft's flight path, they tend to cancel each other such that the subject aircraft flies between them rather than around them (Figure 5). As described earlier, this feature is advantageous in terms of the overall efficiency of the solution, but it requires maneuvering of the obstacle aircraft to avoid conflict with the subject aircraft; i.e., the obstacle aircraft must make enough room for the subject aircraft.
2. Because the random-conflict generator creates

trial trajectories of differing lengths and speeds and then adjusts the aircraft starting times so as to create the maximum congestion, some of the pilots become airborne after others. Thus, if the obstacle aircraft described above have late starting times, the vehicles may have difficulty maneuvering to avoid the subject aircraft.

In other words, the failures shown in Figure 10 result from the fact that the algorithms developed to this point assume that all aircraft are free to maneuver at any time, an assumption that the scenario generator does not abide by. Note that, with a few exceptions, the algorithm was nonetheless able to resolve the bulk of the conflicts successfully. Still, it would be useful to describe how we could handle situations in which the algorithms fail to calculate conflict-free paths.

One approach would determine the exact cause of a failure and attempt to include additional logic to solve the particular failure mode discovered. In this particular case, we might decide that aircraft that have little or no maneuverability and are in close proximity to each other should be considered (from the point of view of other aircraft) as one large object. An example of this approach is the way a person walking through a crowd tends to treat an obvious grouping of people (e.g., a family) as a single large object to be avoided rather than as individuals between which one might maneuver.

Another approach that is both simpler and more robust is the following:

1. Run the algorithms as before.
2. If there are no unresolved conflicts, the processing is completed.
3. If there is a failure, create a large stationary pseudo-obstacle, place it temporarily at the time and position of the conflict such that it is "visible" to one or more of the pilots involved in the conflict, and give the modeled pilots prior knowledge of the upcoming obstacle. (One might imagine that the pilots are told to replan their routes because the area where the conflict was detected has been temporarily declared as restricted airspace.)
4. Return to step 1.

The above approach has the effect of forcing the modeled pilots to find alternative routes when their original routes do not sufficiently resolve a conflict. Also, the approach will resolve a wide variety of algorithm failures regardless of the exact causes of those failures.

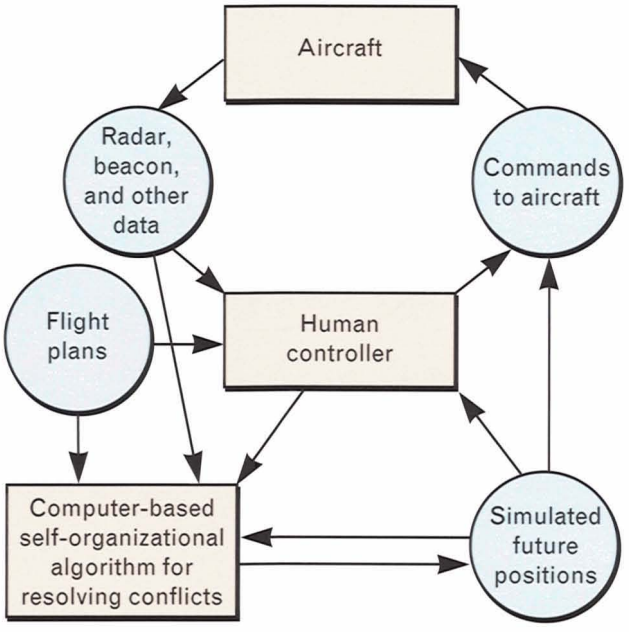


FIGURE 12. Information flow of computer-assisted ATC using a self-organizational algorithm.

Practical Implementation

The actual implementation of computer-assisted ATC that uses self-organizational algorithms could be relatively simple and straightforward (Figure 12):

1. The same information that is provided to controllers, for example, aircraft positions and flight plans, is also supplied to the computer-based system,
2. The computer processes the information, looking for conflicts and developing one or more solutions to any conflicts discovered, and
3. The acceptable solutions are presented to the controller, who may then choose to use any of the computer-generated solutions in addition to any of his or her own solutions.

The mechanics of the second step—the determination of viable flight paths—would follow the general outline below:

1. Analyze traffic and identify aircraft that are new

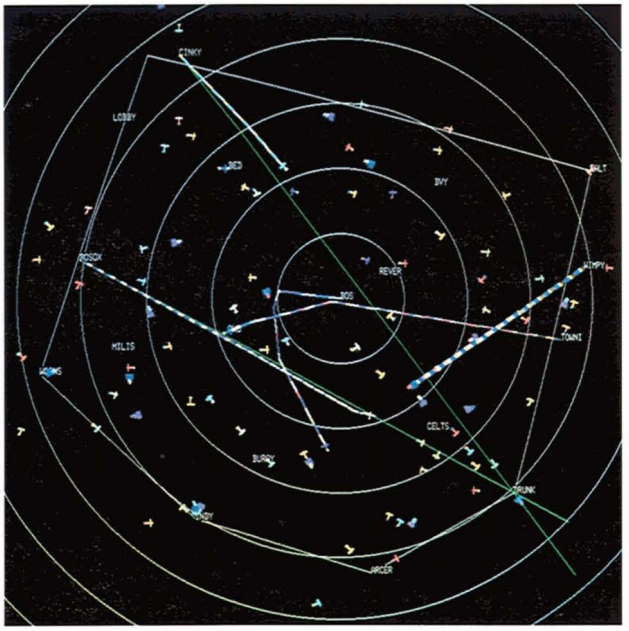


FIGURE 13. Example of ATC sector. The sector is very roughly a square area that is 40 mi × 40 mi. The boundary of the sector is defined by 10 navigational fixes: CINKY, EALT, WIMPY, TOWNI, DRUNK, ARCER, INDY, WOONS, BOSOX, and LOBBY. The airplanes have been color coded with the different colors representing different altitudes. The segment lengths of an airplane's flight path indicate the vehicle's speed.

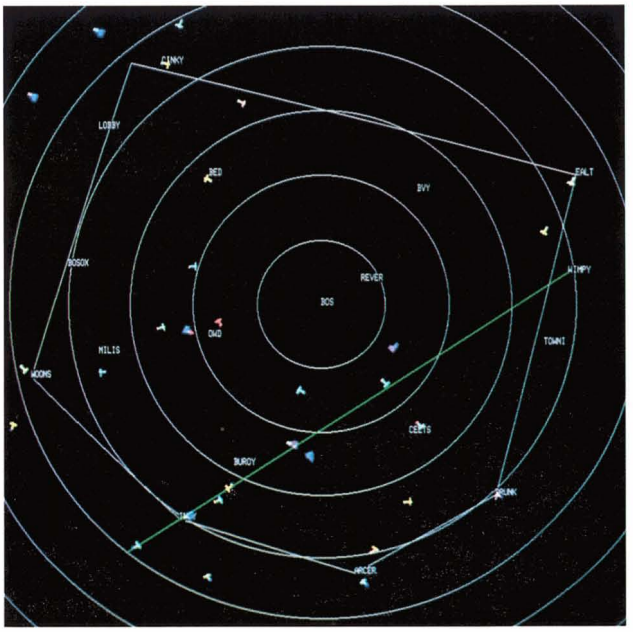


FIGURE 14. ATC simulation sequence: ideal path of subject aircraft entering over the INDY fix.

and/or have significantly deviated from their assigned trajectories. If no such aircraft are identified, no further action is required.

2. Project the ideal paths of the aircraft identified in step 1 and determine which, if any, involve conflicts. If there are no conflicts, no further action is required.
3. Set up the air traffic model with the current live traffic situation and run the self-organizational pilot simulation to determine valid resolutions. The simulation can be run multiple times with varying constraints to generate multiple solutions. For example, one useful constraint would restrict the algorithm from altering the course of an aircraft that has already been assigned a path; that is, the constraint would require that the simulation resolve the conflict without changing the trajectories of aircraft that have already been admitted to the sector.
4. Simplify the flight paths (as described previously) to eliminate as many waypoints as possible and to minimize the path lengths.
5. If appropriate, analyze and assign weightings or rankings to the various solutions prior to their presentation to the controller.

In short, we can use computer-based self-organizational algorithms to solve real-life air traffic conflicts by creating a model of the real traffic situation, operating that model at a speed hundreds to thousands of times faster than real time, and then forwarding the modeled flight paths to the real aircraft.

An ATC Simulation

After the pilot algorithms had been developed and tested in a variety of scenarios and the means by which such algorithms might be applied to live ATC had been outlined, a model was created to test the application of the algorithm in more lifelike situations.

ATC Model

The ATC model defines a sector, which is very roughly a square 40 mi × 40 mi, with its boundary defined by 10 navigational fixes and its interior containing four airports from/to which traffic may be assigned to depart/arrive (Figure 13). The modeled sector may be most accurately described as a Terminal Radar Approach Control (TRACON) sector, which has un-

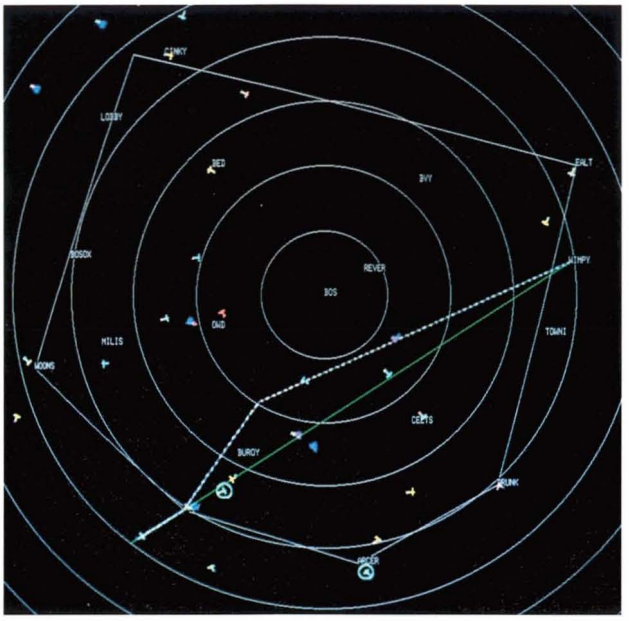


FIGURE 15. ATC simulation sequence: path of subject aircraft selected by algorithm.

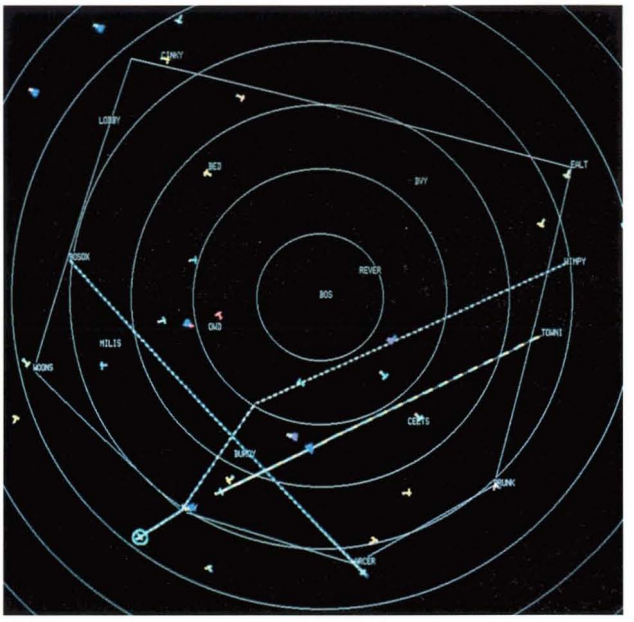


FIGURE 16. ATC simulation sequence: the paths and speeds of two aircraft that are at the same altitude as the subject aircraft. (Note: The longer bars indicate higher speeds.) Aircraft A, just northeast of the INDY fix, is ahead of, moving faster than, and traveling in the same direction as the subject aircraft, and is thus not a source of conflict. Aircraft B, traveling from ARCER to BOSOX, does cross the path of the subject aircraft.

usually large amounts of through traffic.

Flights were introduced into a modeled sector according to a number of criteria:

1. Aircraft were added to the simulation at random intervals. The average interval was variable; thus the traffic load and density were also variable.
2. The bulk of flights were through the sector; that is, the bulk of flights arrived over one of the bordering fixes and exited over another fix some distance away. For the remainder (10% to 20%) of flights, one of the four airports inside the sector was the origination/destination point. The selection of fixes was subject to the constraints of criteria 3 and 4 below. (Note: The algorithms did not require the use of fixes on the borders of sectors any more than they required fixes within sectors. Indeed, one goal of this research was to eliminate the need for any arbitrary fixes. Rather, the purpose of using fixes for entry/exit points was twofold: to demonstrate how the use of self-organizational solutions might be integrated into the current ATC system and to provide a familiar framework for understanding the simulation being performed.)
3. No two flights were introduced such that they conflicted upon entry; that is, there was sufficient time between incoming flights at the same altitude at the same fix. (Note: This criterion is one that any controller requires of the aircraft being passed into his or her sector.)
4. The assignment of exit fixes/altitudes (or destination airports) was monitored and altered as necessary so as not to exceed the capacity of a particular fix/altitude (or airport) to accept traffic. (Note: The model did not include a self-queuing algorithm; that is, the pilots did not know how to create, enter, or exit holding patterns. In the future, such algorithms may be implemented.)
5. Although an aircraft could change its altitude within a sector, through-sector flights had to exit the sector at the same altitude at which they entered. This artificial (and conservative) requirement simplified the software coding of the model.
6. All aircraft that were inbound to an airport were required to align themselves with the

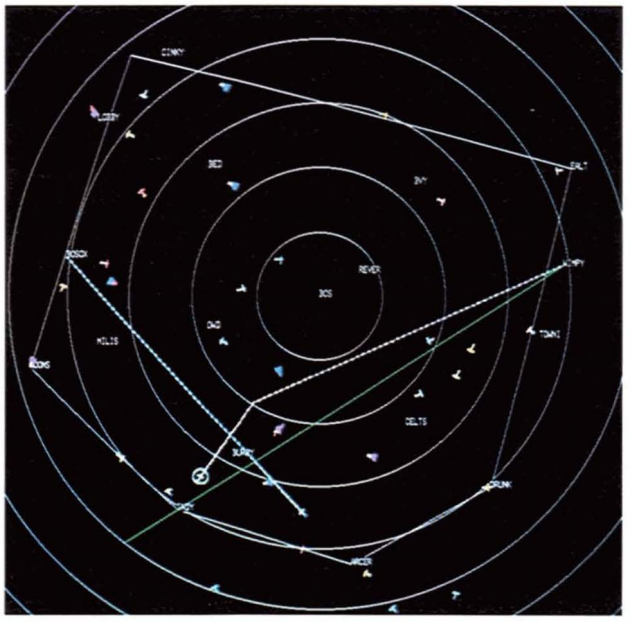


FIGURE 17. ATC simulation sequence: the positions of the subject aircraft and aircraft B shortly after the subject aircraft has entered the sector.

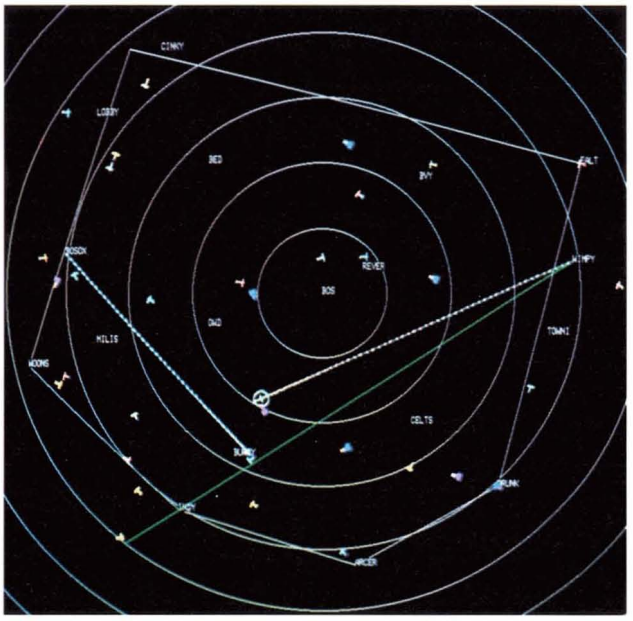


FIGURE 18. ATC simulation sequence: the positions of the subject aircraft and aircraft B at the time of their closest approach to each other.

runway on approach.

7. All aircraft were assumed to be under positive control; that is, visual flight rules (VFR) flights were not modeled.
8. Aircraft were assigned nominal speeds that ranged from 60 kn (the speed of a Piper Cub) to 250 kn.

Results

This subsection presents a series of snapshots that demonstrate the resolution of a conflict between two aircraft.

1. Figure 14 shows the ideal path of an aircraft that is about to enter the modeled sector over the INDY fix.
2. Figure 15 displays the path that the algorithm has computed for this inbound aircraft. Note that the computed path is not the same as the ideal path, indicating the presence of one or more conflicts with the ideal path. In the figure, time/speed is encoded in the lengths of the alternating bars, with longer bars indicating higher speeds. It should be noted that the ATC model does not permit aircraft to begin avoidance maneuvers until they are inside the sector boundaries.
3. Figure 16 shows the paths and speeds of two aircraft that are at the same altitude as the subject aircraft. Aircraft A, just northeast of the INDY fix, is ahead of, moving faster than, and traveling in the same direction as our subject aircraft, and is thus not a source of conflict. Aircraft B, traveling from ARCER to BOSOX, does cross the path of our subject plane, hence the avoidance maneuver calculated by the algorithms.
4. Figure 17 shows the positions of the subject aircraft and aircraft B shortly after the subject aircraft has entered the sector, and Figure 18 shows the two aircraft at the time of their closest approach to each other. Note that the conflict has been resolved.

The above example comes from a run with relatively low traffic densities. Extensive testing was done at three higher traffic rates averaging 35, 55, and 100 aircraft in or entering the sector at any given time. By current standards, these three runs represent traffic densities that range from busy to quite unreasonable. Indeed, the run that averaged 100 aircraft inside the 40 × 40-mile sector at any one time averaged only 7 to 8 sec between handoffs. Figures 19, 20, and 21 contain summary plots for the three traffic densities of 35, 55, and 100 aircraft, respectively.

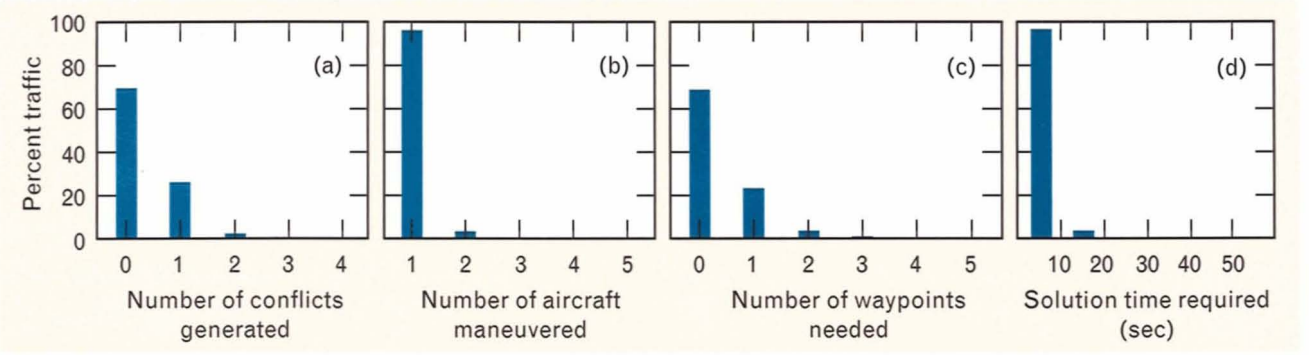


FIGURE 19. Simulation results for scenario involving 35 aircraft: (a) number of conflicts caused by the introduction of new traffic, (b) number of aircraft that had to be maneuvered to resolve the conflicts of part *a*, (c) number of waypoints required in the solution routing paths to resolve the conflicts, and (d) solution time required. Note that nearly 70% of the new traffic did not cause any conflicts, as shown in part *a*. Of the conflicts that were caused by the introduction of new traffic, nearly all could be resolved by maneuvering just the incoming aircraft (i.e., aircraft already in the sector did not have to be redirected), as shown in part *b*. More than 90% of the solution paths did not require more than 1 waypoint, as shown in part *c*, and the solutions could be calculated in less than 10 sec for nearly all the conflicts, as shown in part *d*.

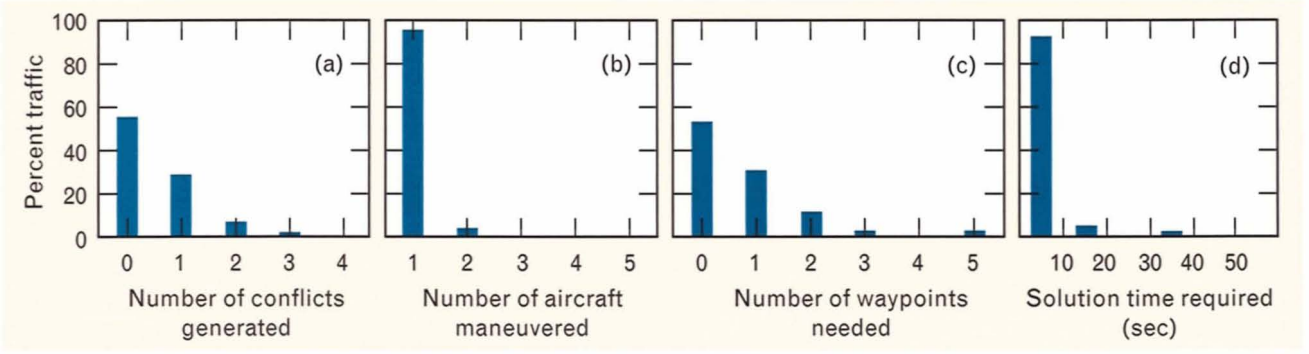


FIGURE 20. Simulation results for scenario involving 55 aircraft: (a) number of conflicts caused by the introduction of new traffic, (b) number of aircraft that had to be maneuvered to resolve the conflicts of part *a*, (c) number of waypoints required in the solution routing paths to resolve the conflicts, and (d) solution time required.

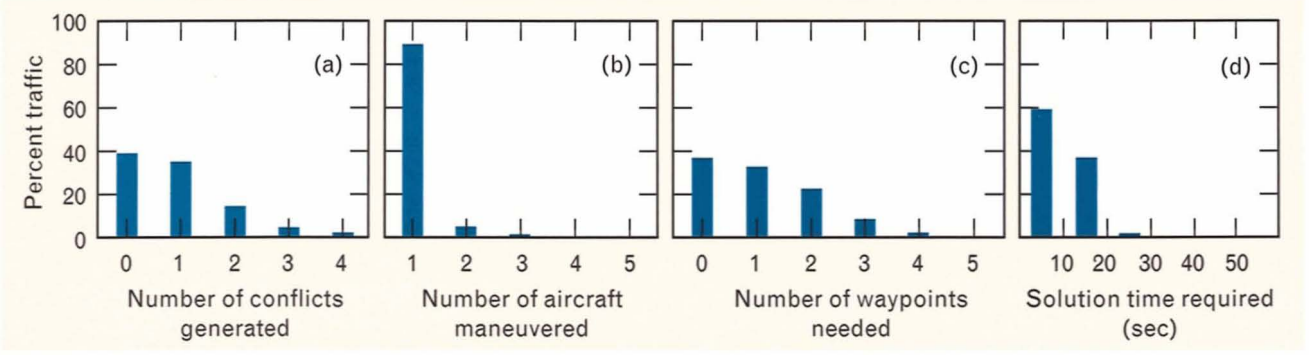


FIGURE 21. Simulation results for scenario involving 100 aircraft: (a) number of conflicts caused by the introduction of new traffic, (b) number of aircraft that had to be maneuvered to resolve the conflicts of part *a*, (c) number of waypoints required in the solution routing paths to resolve the conflicts, and (d) solution time required. Note that more than 60% of the incoming new aircraft create conflicts (as many as four), as shown in part *a*. But the algorithms were basically able to resolve the conflicts by altering the course of just the incoming new aircraft, as shown in part *b*. For the majority of the cases, a time of less than 10 sec was required to resolve the conflict, as shown in part *d*.

Figures 19(a), 20(a), and 21(a) display the number of conflicts generated by the introduction of new aircraft, that is, the probability that the ideal path of an aircraft entering the sector will conflict with the existing flight plans of aircraft already in the sector. Not surprisingly, the likelihood of conflict becomes greater with increasing traffic density: in the 100-aircraft scenario (Figure 21[a]), more than 60% of incoming new aircraft create conflicts (as many as four).

Figures 19(b), 20(b), and 21(b) display the number of aircraft that had to be maneuvered to resolve the conflicts. In even the most dense scenario (Figure 21[b]), the algorithms were able, with a few exceptions, to resolve the conflicts without having to alter the course of aircraft already in the sector. (Note: To approximate the problem solving of real-life air-traffic controllers, the ATC model required that conflicts be resolved by maneuvering just a single aircraft whenever possible. This restriction, which made the algorithm's work more difficult and resulted in less-optimal paths than if other aircraft could also be maneuvered, may be omitted from an operational system.)

Figures 19(c), 20(c), and 21(c) show the number of waypoints that were necessary to avoid conflicts. Even for the highest-density test (Figure 21[c]), resolution of conflicts could usually be avoided through the use of only one or two maneuvers. In the few cases in which more than two waypoints were required, the calculated solution usually also called for the maneuvering of two or more aircraft.

Lastly, Figures 19(d), 20(d), and 21(d) summarize the computational requirements. The ATC simulation, algorithm calculations, and display were all performed in real time on a single Sun 3/160 workstation (technology that is roughly seven years old). Even though the algorithms were not particularly optimized for CPU efficiency, most solutions required less than 3 sec of CPU time for the lowest-density scenario. The highest-density scenario had an average time of less than 15 sec for the calculation of a conflict-free routing.

The ATC simulations summarized in Figures 19 through 21 demonstrate that self-organizational algorithms can provide timely and operationally plausible solutions to ATC conflicts, even at traffic densities much higher than those of today's busiest sectors.

Conclusion

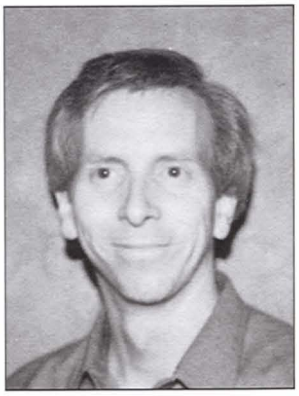
Self-organizational ATC algorithms show great promise as a means for employing computers to resolve ATC conflicts. But such algorithms do not represent a revolutionary technique for efficient ATC. Indeed, the set of visual flight rules (VFR) forms a self-organizing ATC system that has worked well for decades.

The ATC simulation described in this article was of low to moderate fidelity. For example, the traffic generated in the simulations was completely random, whereas air traffic in real life tends to form distinct patterns. Still, the limits placed on the performance of modeled aircraft were conservative and the levels of traffic density and frequency of conflict were extremely demanding. That said, the author is not aware of another ATC technique that can solve, in timely fashion, conflicts of the complexity and traffic density represented by the tests and simulations described in this article.

In comparison with self-organizational approaches, ATC methods based on expert systems have tended to require large amounts of processing capacity and/or clock time to determine efficient solutions for nontrivial conflicts. Thus the basic techniques described in this article could, at a minimum, be used as a pointer or guide to an expert system by rapidly finding one or more trial solutions that the expert system could then use as starting points for its own solutions. Moreover, the results of the ATC simulation indicate that an approach in which a self-organizing model is the central means of conflict resolution would not only be practical but also quite robust and relatively simple to implement.

Acknowledgments

The author thanks the Innovative Research Program at Lincoln Laboratory and David Spencer for his assistance and endorsement of this work. The author is also grateful for the support of James Evans, Mark Weber, David Bernella, and Carroll Edward Schwartz of the Weather Sensing Group, and Carl Nielsen, Jr., John Fielding, and Richard Dennis of the Surveillance and Control Division.



MARTIN S. EBY
is president and owner of Source Code Systems in Wichita, Kansas. He was previously with the Weather Sensing Group at Lincoln Laboratory, where his focus of research was on ATC aids. Martin received a B.S. degree in chemical engineering from Kansas State University and an M.S. degree in chemical engineering with emphasis on control systems from the University of California, Santa Barbara.