Published at the 13th IFAC Symposium on Automatic Control in Aerospace, Palo Alto, California, Sept., 1994.

THE FINAL APPROACH SPACING TOOL

T. J. DAVIS*, K. J. KRZECZOWSKI, and C. BERGH*****

**NASA Ames Research Center, Moffett Field, California 94035*
***Sterling Federal Systems, Palo Alto, California 94303*
****MIT Lincoln Laboratory, Lexington, Massachusetts 02173*

Abstract. A system for assisting terminal area air traffic controllers in the management and control of arrival traffic, referred to as the Final Approach Spacing Tool (FAST), is being developed at NASA Ames Research Center. In a cooperative program, NASA and the FAA have efforts underway to install and evaluate the system at the Dallas/Fort Worth Terminal Radar Approach Control facility beginning in 1994. This paper will review the algorithmic components, the human-machine interface, and the results of recent simulations.

Key Words. Air-traffic control; Automation; Computer Application; Heuristic programming; Simulation.

1. INTRODUCTION

The development of automated systems for the control of air traffic has long been the objective of researchers and engineers. The continued growth of air traffic nationwide has caused increases in air traffic delays and has put considerable stress on both existing air traffic control systems and on the air traffic controllers. This paper describes the design of an automation system for assisting controllers in the management and control of arrival traffic in the terminal area.

Early work in the automation of terminal air traffic control was presented in the late 1960's (Martin and Willet, 1968). This system provided speed and heading advisories to controllers to help increase spacing efficiency on final approach. Although traffic tests of the system showed an increase in landing rate, controllers found that their work load was increased and rejected the system. An examination of the concept suggests that while some aspects of the design were sound, its acceptance was limited by the technology of the time period, especially the lack of an adequate controller interface. More recently, several automation systems have found their way into operational use in Europe due in large part to the introduction of modern computer processing and interfaces, and because of more careful design approaches (Volckers, 1990; Garcia, 1990). However, these systems do not contain detailed modeling of complex runway operations. In addition, recent fast time simulation studies have confirmed the potential for increasing landing rates with the assistance of active advisories for controllers in the terminal area (Credeur and Capron, 1989).

A candidate system for the automated management and control of terminal area traffic, referred to as the Center TRACON Automation System (CTAS), is under development at NASA Ames Research Center in collaboration with the FAA's Terminal Air Traffic Control Automation Program Office. The elements comprising the CTAS are the Traffic Management Advisor (TMA), the Descent Advisor (DA), and the Final Approach Spacing Tool (FAST) (Erzberger *et al*, 1993). The advisories generated by these tools assist controllers in handling aircraft arrivals starting at about 200 n.mi. from the airport and continuing to the final approach fix. Recently, the elements of the CTAS system have been evaluated in a series of real-time simulations at NASA Ames Research Center and in field testing at facilities serving the Denver and Dallas/Fort Worth areas.

This paper focuses on the design and implementation of the terminal area portion of CTAS referred to as FAST. The main function of FAST is to provide landing sequence, landing runway assignments, speed, and heading advisories that help controllers manage arrival traffic and achieve an accurately spaced flow of traffic on final approach. The paper concludes with a description of current laboratory and field testing results.

2. FINAL APPROACH SPACING TOOL

FAST consists of several major software components: a route analyzer and trajectory synthesizer, a sequencer and scheduler, a conflict resolver, a runway allocator, and a controller interface. Each of these components is described below along with examples illustrating their operation.

2.1. Route Analysis and Trajectory Synthesis

The FAST system is dependent on the accurate estimation of arrival times for all aircraft. These arrival times are used by FAST for sequencing and scheduling aircraft to the runway threshold. The process within CTAS that is responsible for the rapid update and accurate calculation of estimated times of arrival (ETA's) based on radar track or flight plan data is

referred to as the Route Analysis and Trajectory Synthesis (RA/TS) program.

The set of ETA's that the RA/TS computes represents ranges of possible arrival times given an aircraft's predicted route of flight combined with possible variations in degrees of freedom along those routes. Typical degrees of freedom include speed, horizontal and vertical maneuvers. Upon receipt of a flight plan or radar track data update (x, y, altitude, ground speed), the RA portion of RA/TS 'categorizes' each aircraft's situation for each potential landing runway in terms of destination airport, airport configuration, geographical sections of airspace, engine type (jet, turbo-prop, piston), approach segment (downwind, final, base, etc...), and aircraft states (x, y, altitude, heading, speed). Each situation category has a name and a complete description of route/degree of freedom combinations that are possible for the aircraft. The RA uses this site-adaptable data for each situational category to build a series of one or more routes for each aircraft, apply degrees of freedom to those routes, and finally to request the TS portion of RA/TS to compute ETA's for each route/degree of freedom combination.

The trajectory synthesis (TS) portion of RA/TS is a modified version of an algorithm originally designed for computing descent-from cruise trajectories (Erzberger and Tobias, 1986). The inputs to the TS program are the aircraft state, winds aloft, temperature and pressure profiles, a series of waypoints depicting the expected route of flight for an aircraft, and vertical and speed constraints on the predicted route. The outputs from the TS include a complete time-based (4D) trajectory along the expected path including all pertinent data for resolving conflicts and estimating times of arrival at points along the path.

As the aircraft flies through the arrival airspace and descends to the runway, it will change situation categories as it transitions from one flight segment to the next, producing stable sets of ETA's. These sets of ETA's, form the basis for the sequencing and scheduling process. Once the sequencing and scheduling process is completed, the same set of RA/TS trajectories will be used for conflict resolution. Finally, they will be used as a reference in computing expected delay for aircraft in the runway allocation process.

2.2. Sequencer and Scheduler

Human controllers have the ability to construct a plan about how aircraft will merge together and land safely. Though we do not understand the controller's cognitive process in making such plans, a working model hypothesizes that they do this by comparing an aircraft's projected position to the projected position of other relevant aircraft. An automation aid will have to make the same comparisons. The method of breaking an aircraft's predicted trajectory into trajectory segments and grouping them with similar trajectory segments of other aircraft allows the comparisons to be limited to only the aircraft that have an effect on each other.

A trajectory is made up of a set of time steps at defined intervals. Each time step contains a predicted x, y, altitude, speed, and heading of an aircraft at a future time. A trajectory segment is a grouping of time steps that fall within a predefined segment of flight. Figure 1 shows an aircraft and its trajectory broken into four trajectory segments called LONG_LEFT, DOWNWIND_LEFT, BASE_LEFT, and FINAL.

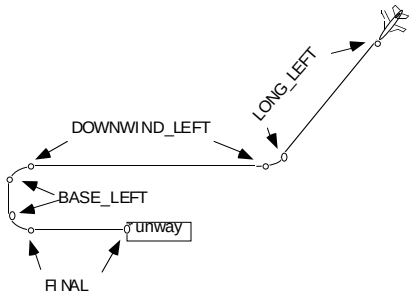

Fig. 1. Trajectory Segments for an aircraft on long left

With trajectories broken into common trajectory segments, the FAST planning process consists of comparing only the like trajectory segments. For example, all DOWNWIND_LEFT segments for different aircraft will be compared. This method eliminates the need to compare all aircraft trajectory segments, and entire aircraft trajectories.

The first step in producing a plan or sequence is to determine an order in which to land the aircraft. It was learned, through extensive real-time simulations, that to produce an acceptable sequence for controllers to follow, it was necessary to consider all merges within the airspace, not just the merge on the final approach course. Ordering is the process of both creating a relative sequence within each trajectory segment and combining those sequences into a consistent global ordering for each runway.

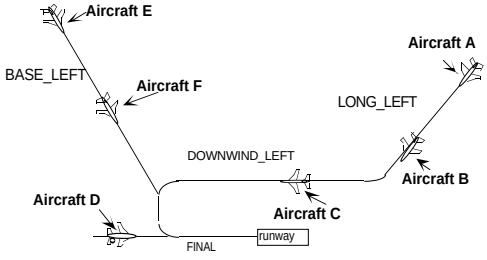

Fig. 2. Sequencing example

Figure 2 depicts a situation where six aircraft merge together to land on the same runway. The network of trajectory segments can be thought of as a tree, Fig. 3. The tree branches represent merging possibilities that can take place within the TRACON, and the leaves on

each branch are in the relative order of the aircraft on their current trajectory segment.

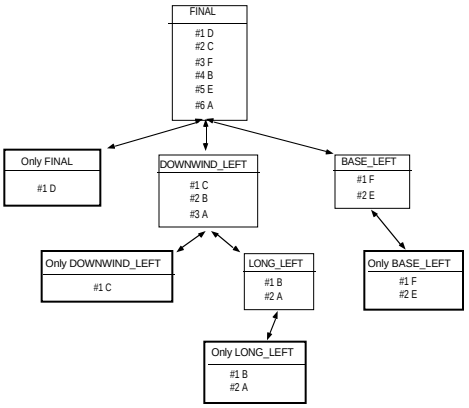

Fig. 3. Trajectory Segment tree

The sequencing starts at the last leaf of each branch in Fig. 3. Once the aircraft in the leaves have been ordered, FAST blends these relative sequences of the branches into the final sequence. The relative order of the aircraft is maintained throughout the tree.

At the beginning of each sequencing cycle, FAST builds new trajectories for updated aircraft positions. The sequencing algorithm uses both these new trajectories and the previous sequence to calculate a new sequence. There are two ordering algorithms which are utilized, one which orders the leaves of the tree up to the final merge, and one which accomplishes the final merge.

The first algorithm makes use of a general sorting function that accepts as input an unordered list, and a function which compares two members of that list, returning their relative order. The heart of the algorithm is a sequence order function which makes the comparison for two aircraft. The function searches the list of time steps associated with the trajectory segment it is currently ordering, to find the earliest instant within the segment that two aircraft have time steps. These two time steps are called the First Common Time Steps (FCTS).

The primary input to the logic is a measure of how far ahead/behind one aircraft is to another. A distance is calculated from the FCTS to the end point of a trajectory segment for each aircraft . The difference in the distances, divided by the required separation for any two aircraft gives a Normalized Separation Distance (NSD).

$$NSD = \frac{(\text{distance B} - \text{distance A})}{\text{Required Separation}} \qquad (1)$$

In Eqn. 1, if the NSD is positive, aircraft A would be ahead of aircraft B; a negative value would indicate that aircraft A is behind aircraft B. The exact value measures how much ahead/behind A is, relative to B. The required separation is defined by the aircraft weight classes and is shown in Table 1.

Table 1. Required separation (n.mi.)

Leading Aircraft Type	Trailing Aircraft Type		
	Heavy	Large	Small
Heavy	4	5	6
Large	3	3	4
Small	3	3	3

The remaining inputs to the logic are: the distance from each aircraft's current location to the specific trajectory segment being ordered, the speed difference between aircraft at the FCTS, and the last updated or previous relative order. If there is no previous order, FAST bases its ordering decision on the sign of the NSD. However, if the value of the NSD is small (less than 0.5), and one aircraft is flying significantly faster than the other, the faster aircraft will be ordered ahead.

When a previous order exists, the system no longer bases the sequencing decision on the sign of the NSD. In order for the sequence to change, the aircraft that was previously sequenced behind must pull ahead of the lead aircraft by a specified NSD. The amount it must pull ahead is a function of the inputs to the logic. Figure 4 displays a family of curves that describe the decision region for determining if the previous order should be changed. The shape of the curve was experimentally derived from real-time simulations with air traffic controllers.

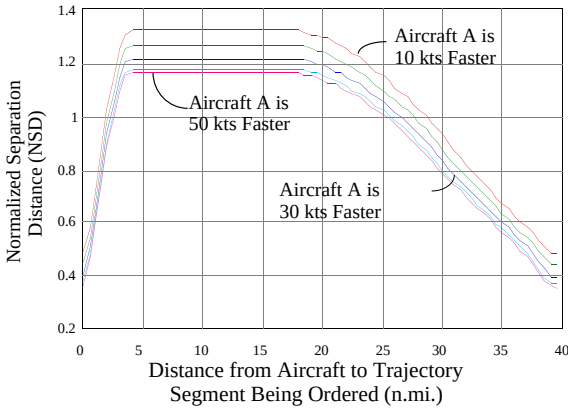

Fig. 4. Sequencing order function for merges not on final approach.

The graph defines two decision regions, parameterized by speed difference, for the case where aircraft B was sequenced ahead of aircraft A during the last update cycle. A positive NSD is a measure of how much aircraft A is ahead of aircraft B for this new update cycle, and a positive speed difference is a measure of how much faster aircraft A is moving than aircraft B. The curves are loci of constant speed difference in knots. If the updated input for the two aircraft results in

a point plotted above their speed difference line, the previous relative order is reversed. If the update results in a point plotted below the line, the previous sequence is maintained.

Reading the graph from right to left, the curve climbs from an NSD of almost zero to one between a distance from the trajectory segment of forty to twenty miles. This allows FAST to be more responsive to sequence reversals when the aircraft are farther away from a trajectory segment and less responsive closer to a segment. As the aircraft approaches and joins the trajectory segment, FAST will quickly respond to changes in sequence. This is reflected in the curve, which drops back to nearly zero again in the last few miles.

The final merge algorithm merges streams of aircraft onto the FINAL trajectory segment in a similar manner by taking advantage of the additional time information that the RA/TS provides about the runway threshold arrival times (time range). The input to the algorithm is a set of ordered lists of aircraft. The output is a single ordered list that contains all the aircraft. The final merge algorithm compares the first aircraft on each list, two at a time, and determines which aircraft is first. It then removes this aircraft from its input list, adds it to the output list, and starts the process all over again. Part of the comparison is a calculation of a Scheduled Time of Arrival (STA). The STA for the number one aircraft is set to the aircraft's nominal arrival time. STA's for the remaining aircraft are calculated by Equation (2).

$$\text{STA of Trail Aircraft} = \underset{\text{Max}}{\text{Arg}} \left\{ \begin{array}{r} (\text{STA of lead aircraft} + \\ \text{requited separation}): \\ \text{Aircraft's Nominal ETA} \end{array} \right\} \quad (2)$$

Consequently, as aircraft are added to the output list, they may need to absorb delay in order to be sequenced behind aircraft already in the list.

The algorithm computes a delay for each aircraft by subtracting the STA from the aircraft's nominal time. It compares two aircraft, A and B, by summing the delay that A and B will incur when they are ordered behind the last aircraft in the output list . Both possible sequences are checked: A followed by B and the reverse order, B followed by A. The primary input to the logic is the difference in these sums. The algorithm chooses the order based on a compromise between reducing total delay and disturbing the established order.

Figures 5 and 6 display the decision region for the final merge algorithm. Figure 5 describes the part of the decision region when at least one of the two aircraft has enough delay capability to fit behind the other. Figure 6 describes the part of the decision region when both aircraft are almost out of delay capability. As mentioned earlier, these curves were experimentally derived in real-time simulation.

In Fig. 5, the y-axis denotes the delay savings when choosing the reverse order. The x-axis denotes the path distance of the trailing aircraft to the FINAL. The lines on the graph represent the loci of constant percentage of the delay required in the previous order for the trailing aircraft to fit behind the lead aircraft. As described before, if the updated input results in a point plotted above its delay line, the previous order is reversed. If the update results in a point plotted below the line, the previous order is maintained. As the percentage of delay the trailing aircraft is able to produce in order to remain behind the lead aircraft decreases, FAST is more likely to reverse the order. In fact, if it is unable to produce better than 105% of the delay required, then the order will change without any time savings.

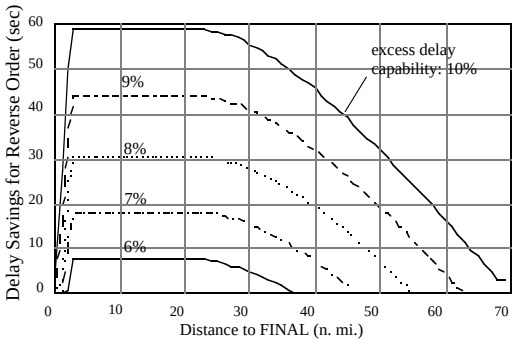

Fig. 5. Sequence order function when at least one aircraft has enough delay to fit behind the other

FAST is more responsive when the aircraft are far away from or nearly on the trajectory segment, and less responsive as the aircraft are approaching the trajectory segment.

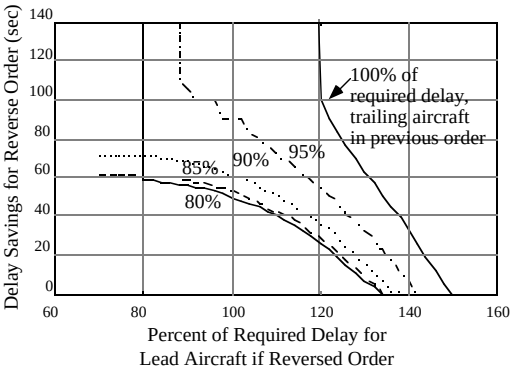

Fig. 6. Sequence order function when both aircraft are running out of delay

Figure 6 represents the decision space when neither aircraft can absorb all of the necessary delay. Recall that the aircraft being compared by this section of the decision space are competing for a slot behind aircraft that have already been ordered on the output list . The y-axis is delay savings by choosing the reverse order instead of the previous updated order. The x-axis is the percent of required delay for a given sequence that a lead aircraft in the previous order is capable of producing if the previous order were reversed. The lines on the graph are loci of constant percentage of the

required delay that a trailing aircraft is capable of producing for the previous order. As described before, if the updated input results in a point plotted above its delay curve, the previous order is reversed. If the update results in a point plotted below the curve, the previous order is maintained.

As the aircraft in front runs out of delay capability to fit behind the aircraft that is previously sequenced behind it (moving left along the x-axis) FAST becomes less likely to reverse the order. In fact, if the trailing aircraft has 100% or more of the delay required to stay behind, it will become impossible for the sequence to change as the aircraft ahead runs out of delay capability.

2.3. Conflict Resolution

All trajectory segments for an aircraft are checked for conflicts with other aircraft within the same segments. If there are no conflicts for an aircraft, it will be assigned its nominal trajectory. When a conflict is predicted, one or both aircraft trajectories must be manipulated to resolve the conflict. Because the aircraft are already ordered by FAST, the system knows which aircraft is ahead and which aircraft is behind. The algorithm will add delay to the trailing aircraft in order to resolve the conflict. The system accomplishes this by searching the trajectory for degrees of freedom which will help to resolve the conflict. The magnitude of the conflict is measured and translated into a required delay for the aircraft before it reaches the conflict point. Because the system knows which degrees of freedom will help to resolve the conflict, it can combine this knowledge with the route/degree of freedom/ETA values it received from the RA/TS to bound and then begin the iterative process for resolving the conflict.

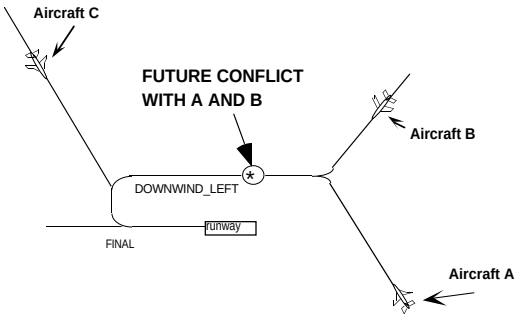

Fig. 7. Conflict on DOWNWIND_LEFT

The process of resolving conflicts contains a number of complicated situations. It may seem that for each aircraft added to the sequence, FAST only has to resolve violations that are with an aircraft ahead in the final sequence. Unfortunately, there are a number of cases where the situation is more complicated. For example, Fig. 7 shows an aircraft A that will be merging with another aircraft B on DOWNWIND_LEFT and then another C on FINAL. Assume that the final ordering is B, C, then A. The idea of resolving only the conflicts A had with C could leave a DOWNWIND_LEFT conflict unresolved. The problem with checking for conflicts only with the aircraft ahead on FINAL is that a merge could be missed. FAST will search for and resolve conflicts on all trajectory segments between a given aircraft and the aircraft sequenced ahead of it.

2.4. Runway Allocation

An algorithm for allocating runways based on a cost function to minimize delay was described by Brinton (1992). However, the problem of achieving a both procedurally acceptable and efficient runway allocation solution is beyond the reach of real-time optimization and instead requires detailed knowledge of a facility's procedures.

Before the algorithm employed in FAST is described, a brief description will be given of how a controller selects a default or preferred runway, and how controllers select aircraft to be switched to secondary runways. In general, aircraft are vectored from Center airspace into a TRACON over a feeder gate or metering fix. Aircraft engine types (e.g. turbo-jets, turbo-props, piston) and feeder gate assignment map to a preferred runway which is typically the closest runway to that feeder gate. Depending on the procedures at a given TRACON, an aircraft may be eligible to change to secondary or alternate runways. Situations which would influence a controller-initiated runway change include excessive or unbalanced delay buildup on the preferred runway, controller workload for a given runway, and airline or control tower preferences for ease of ground traffic movement. A controller will select which aircraft to change to an alternate runway based on a number of considerations such as separating aircraft of a dissimilar engine type or weight class from the other aircraft in a busy stream of traffic, avoiding potential conflicts in current streams of traffic, or avoiding potential conflicts in merging streams of traffic. Ideally, a controller would like to change the runway early in the traffic flow (i.e. near the feeder gates), but because of the uncertainties of making such a decision early in the flow, changes are commonly held off until the last possible moment. This can cause an undesirable increase in workload for the pilots of arriving aircraft because of the late changes in selecting navigation frequencies and configuring the aircraft for an approach.

The strength of an automation system such as FAST is its ability to assign runways based on accurate estimations of delay savings and workload benefits at an early stage of the arrival process. The runway allocation algorithm employed in FAST attempts to meet three primary objectives: 1) making an early and accurate decision, 2) reducing overall system delay and 3) maintaining controller acceptability. The algorithm is heuristically-based and site-adaptable. The approach is to define the preferred runway for all aircraft in the landing sequence and then to select the set of all aircraft which are eligible for reassignment, apply criteria to

narrow this set to a most likely aircraft to be reassigned, to test the aircraft's new runway in a full sequencing and conflict resolution cycle with all other aircraft, and finally to apply detailed criteria to this solution set for all aircraft.

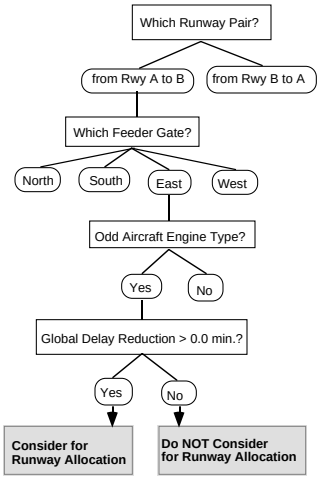

Fig. 8. Example of decision tree for selecting aircraft for runway allocation

The set of aircraft eligible for runway reassignment is defined by a runway allocation time window for each runway. The time window begins with a "start testing runway allocation time horizon" measured in expected flight minutes from a given runway and ends with a "freeze runway allocation time horizon" also measured in expected flight minutes from the runway. Any aircraft with an estimated time of arrival for a runway which falls within the runway allocation time window are deemed eligible for allocation to that runway. Once the set of eligible aircraft are determined for all arrival runways, the system builds an estimated schedule and its associated delays for each aircraft to their currently assigned runway and to any available alternate runways. The system then selects those eligible aircraft which pass a set of runway allocation heuristics. This selection process is based on a site adaptable decision tree file which incorporates facility procedures, delay reduction, and controller heuristics. A simplified example of a runway allocation decision tree is shown in Fig. 8. In this example, only one thread through a series of branches on the decision tree is shown. The tree first branches on runway pair, followed by arrival feeder gate, followed by a criterion labeled "Odd Aircraft Type." This criterion examines the aircraft together with all aircraft meeting the previous criteria (runway pair and feeder gate), and determines if the aircraft currently traversing the decision tree is an odd type (e.g. the only turbo prop in a stream of jet traffic). If this is true, then the system examines a system-wide or global delay reduction criterion. Because the aircraft in this example is an odd engine type in its stream, the delay reduction criterion is small (0 minutes). If we examined the branch on the "No" answer for "Odd Aircraft Type," we would find that the global delay reduction criterion would require a larger value (typically 2-4 minutes). The reason for the difference in delay reduction requirements on these two branches is to force FAST to favor pulling a dissimilar engine or weight class aircraft out of the traffic stream. This serves to reduce workload for the controller.

After all eligible aircraft have passed through this decision tree and thus narrowing the list of all eligible aircraft to a smaller set, FAST then selects a single aircraft which appears to have the greatest delay benefits to the overall arrival system. In some cases, there may not be any aircraft which pass these criteria and in this case, the runway allocation algorithm will not consider any aircraft for that update cycle. Once an aircraft is selected, it is then placed in an alternate runway sequencing and conflict resolution cycle. The entire arrival airspace sequencing problem is solved with this aircraft placed on its alternate runway. This allows the software to evaluate all aspects of the particular runway allocation. Full trajectory solutions are obtained for each aircraft which in turn give accurate sequences, expected delay, and conflict detection for the entire airspace. At this point, a new and more detailed set of criteria are applied. These criteria examine trajectory based issues such as potential conflict resolution problems and exhaustion of critical degree of freedom limits. They are applied to the alternate solution set in order to make the final determination as to whether or not to change the aircraft to the alternate runway.

Once an aircraft has been switched away from a given runway, that runway is blocked off from further consideration for that aircraft. A more optimal solution would be to allow allocation of this aircraft back to its original runway if a situation warrants, but this was found to be unacceptable to controllers. Controllers always have the final authority in the runway assignment, and if a controller directs FAST to assign a given aircraft to a runway through a keyboard entry, the system will freeze that decision and no longer consider that aircraft for any other runway. Finally, once an aircraft's ETA falls below a runway's freeze time horizon, that runway will be blocked off from further consideration. After all but one runway has been blocked off, the runway assignment advisory is frozen for the remainder of the flight. In nearly all cases, the aircraft has a frozen runway assignment before twelve minutes of flight time from the runway.

2.5. Controller Interface

The development of the controller interface has focused on implementing FAST on two different controller interface platforms. The first interface platform is the current controller interface in operation at Dallas/Fort Worth called the Full Digital ARTS Display (FDAD). The FDAD's employ a monochrome digital display with trackball, keyboard and analog input devices. The FDAD's will be used as the controller interface in the initial field implementation of FAST. The second interface platform is a Sun workstation color monitor with mouse/trackball and keyboard input devices. This

color workstation was the initial development platform for FAST and was used primarily before the FDAD's became available for testing at NASA.

To assure an effective controller interface, the FAST development team used active air traffic controllers throughout the interface design and four key guidelines evolved: 1) minimize screen clutter, 2) associate advisories with aircraft, 3) minimize keyboard entries, and 4) use graphical advisories where possible.
The output of the previously described algorithmic components produce a set of advisories which must be transferred to the controller interface. There are two primary methods for displaying this information to the controller. The first method is to add information to the aircraft's flight data block. Figure 9 shows a typical flight data block for an aircraft currently displayed in TRACONs.

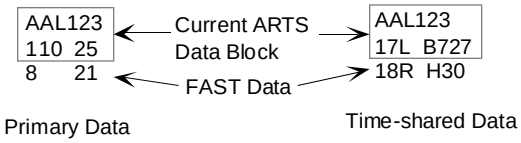

Fig. 9. ARTS flight data block with FAST enhancements

The first line indicates the aircraft identification or call sign. The second line contains two data fields. The first data field contains the current reported altitude (in hundreds of feet) time-shared with a facility scratch pad (typically containing the current runway assignment), and the second data field contains the aircraft's current ground speed (in tens of knots) time-shared with the aircraft type (e.g. Boeing 727 is displayed as "B727"). The third line shown in Fig. 9 is the FAST information data line and is not currently displayed operationally in TRACONs. This line also contains two data fields. The first data field contains the FAST relative sequence number to the runway time-shared with the FAST runway assignment advisory. The second data field can contain both indicated airspeed and heading advisory information. If the data is indicated airspeed information, it is shown in tens of knots, and if it is an advised heading, it is show in tens of degrees (magnetic North) preceded by an "H" for heading.

The second method of advisory display in FAST is graphical and applies to speed and heading advisories. Speed advisories are typically displayed as a marker on the display and an advised indicated airspeed in the third line of the data block as described above (see Fig. 10). These speed advisories are displayed as orange markers along with an orange alphanumeric for the speed value in the data block.

Heading advisories are displayed as a location to issue the turn (shown by a graphical marker), a magnetic heading in degrees next to the marker, and a turn arc depicting the projected aircraft path taking into account its speed, heading and the winds aloft (see Fig. 10). The graphical data is color coded based on arrival feeder gate for the aircraft. The aircraft flight data block changes color to match the graphical advisory. When the aircraft executes or passes the advisory, the flight data block reverts back to its nominal color.

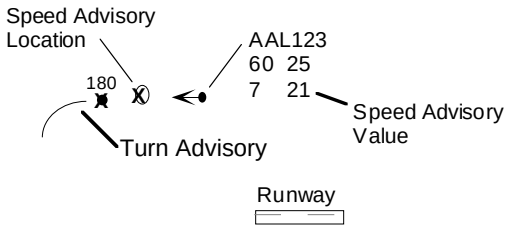

Fig. 10. FAST speed and turn advisory graphics

3. SIMULATION AND FIELD TESTING

The planning and development for field testing and implementation has been ongoing for several years. Recently, the simulations have been conducted almost exclusively with controllers from the Dallas/Fort Worth TRACON in preparation for field testing at that facility.

The simulations have focused on a number of issues ranging from validation of the algorithms to an evaluation of human factors issues. They have assisted both in the development of the system and the planning of the field deployment. The simulations had the following objectives: 1) to assess the potential benefits of FAST, 2) to evaluate controller acceptability, and 3) to develop the system for operational testing.

Initial information on the potential benefits was obtained in a simulation evaluation of FAST operating on a color display in a single runway, Instrument Flight Rules (IFR) configuration (Davis *et al*, 1991). This simulation demonstrated efficient use of airspace, increased landing rates, and controller acceptance of the system. Similar results were obtained in an independent study (Credeur *et al*, 1993).

More recently, simulations of FAST designed for the Dallas/Fort Worth TRACON have demonstrated the system's performance in more complex operations with multiple runways in both IFR and Visual Flight Rules (VFR). These simulations have included parallel simultaneous and staggered approaches, as well as converging approaches. The simulations have been conducted on the FDAD's with traffic scenarios based on live traffic samples from the Dallas/Fort Worth TRACON. The primary results of the tests are discussed below.

First, the controllers have reported that detection of the speed and heading advisories on a monochrome display is difficult and requires additional workload. This is largely because of screen clutter from non-arrival air traffic. In addition, controllers sometimes have difficulties associating advisories with the correct aircraft on the monochrome display. The controllers

stated that the color display mitigates these problems substantially.

Second, the controllers feel that sequence numbers and runway assignment advisories will provide substantial benefits even on the monochrome displays. The controllers report that these advisories often improve on their own decisions. The best use of the sequence numbers is in sectors where the controllers are merging streams of traffic. Runway assignment advisories have been found to match or improve the controller decisions in most cases. Occasionally, controllers have a tendency to doubt runway allocation advisories because FAST can "see" aircraft that are out of the controller's view and thus make an accurate assessment at an earlier stage. However, the "doubtful" runway assignment advisories are nearly always shown to be correct. One can assume that controllers will pass through an adjustment period in order to gain confidence or trust in the system.

4. CONCLUDING REMARKS

An automation system for assisting terminal area air traffic controllers in efficiently managing and controlling arrival traffic has been developed and tested in simulation. The automation system, referred to as the Final Approach Spacing Tool (FAST),was developed through thousands of hours of real-time simulations with active air traffic controllers. Results from the simulations show potential benefits in efficient airspace utilization, reduced controller workload, and increased runway capacity. Some potential risks still remain, primarily in gaining acceptance of the monochrome version of the human interface and for controllers to reach a comfort level with an unfamiliar system. The system is currently undergoing its final phase of development in preparation for field testing at the Dallas/Fort Worth TRACON. The field testing is scheduled to begin in mid-1994 and will include a phased deployment schedule which delivers subsets of the full FAST functionality in increments. The field testing will include periods of shadow control and observation, simulation activities in the Dallas/Fort Worth TRACON training room, and limited operational testing with live traffic.

The FAST system is designed to operate either independently or in direct coordination with the other CTAS tools. However, simulation results indicate that the arrival air traffic management process will receive the most benefits by utilizing all tools sets in CTAS. Integrated tests of all CTAS tools are planned for the Dallas/Fort Worth and Denver airports within the next two years.

5. REFERENCES

Brinton, C. R., "An Implicit Enumeration Algorithm for Arrival Aircraft Scheduling," presented at the 11th Digital Avionics Systems Conference, Seattle, WA, Oct. 1992.

Credeur, L., and Capron, W. R., "Simulation Evaluation of TIMER, a Time-Based, Terminal Air Traffic, Flow Management Concept," NASA TP-2870, Feb. 1989.

Credeur, L., Capron, W. R., Lohr, G. W., Crawford, D. J., Tang, D. A., and Rodgers, W. G., "Final-Approach Spacing Aids (FASA) Evaluation for Terminal-Area, Time-Based Air Traffic Control," NASA TP-3399, Dec. 1993.

Davis, T. J., Erzberger, H., Green, S. M., and Nedell, W., "Design and Evaluation of an Air Traffic Control Final Approach Spacing Tool", *Journal of Guidance Control and Dynamics*, Vol. 14, No. 4, July-August 1991, pp. 848-854.

Erzberger, H., and Tobias, L., "A Time-Based Concept for Terminal Area Traffic Management," Proceedings of the 1986 AGARD Conference, No. 410 on Efficient Conduct of Individual Flights and Air Traffic, Brussels, Belgium, 1986, pp. 52-1-52-14.

Erzberger, H., Davis, T. J., and Green, S. M., "Design of Center-TRACON Automation System", Proceedings of the AGARD Guidance and Control Panel 56th Symposium on Machine Intelligence in Air Traffic Management, Berlin, Germany, 1993, pp. 11-1-11-12.

Garcia, J., "MAESTRO - A Metering and Spacing Tool," Proceedings of the 1990 American Control Conference, San Diego, California, pp. 501-507.

Martin, D. A. and Willet, F. M., "Development and Application of a Terminal Spacing System," Federal Aviation Administration, Rept. NA-68-25 (RD-68-16), Aug. 1968.

Volckers, U., "Arrival Planning and Sequencing with COMPAS-OP at the Frankfurt ATC-Center," Proceedings of the 1990 American Control Conference, San Diego, California, pp. 496-501.