Research Task Order 5B (RTO5B):
Free Flight Simulation Components Development - Hazard Avoidance Planner

Final Report
May 28, 1999

Prepared for:
NASA Langley Research Center
Contract Number: NAS2-98001
Technical Monitor: Monica F. Hughes

Prepared by:
Honeywell Technology Center
3660 Technology Drive
Minneapolis, MN 55418

TABLE OF CONTENTS

1.0 Summary ____ *2*

2.0 Problem ____ *2*

3.0 Introduction ____ *3*

4.0 Technical Approach ____ *4*

4.1 Concept Overview ____ **4**

4.2 Software Overview ____ **8**

4.3 Software Description ____ **10**

4.3.1 User Interface Functions ____ 11
4.3.2 Pseudo-Aircraft Functions ____ 12
4.3.3 Graphics and Display Functions ____ 12
4.3.4 HAP Functions ____ 13

4.4 Simulation - User Procedures ____ **28**

4.4.1 User Overview ____ 28
4.4.2 Development Platform, Hardware and Additional Software Requirements ____ 28
4.4.3 Compilation of HAP Application ____ 29
4.4.4 Program Options Set at Compile Time ____ 29
4.4.5 Database ____ 30
4.4.6 Menu Bar ____ 31
4.4.7 Display Window Content ____ 32
4.4.8 Data Window Content ____ 33
4.4.9 Standalone HAP ____ 33
4.4.10 FASTWIN/HAP ____ 33
4.4.11 Software Installation ____ 34

5.0 Issues and Proposed Improvements ____ *37*

5.1 ADS-B Issues ____ **37**

5.2 Extension of HAP to Full 4-D (Climb, Cruise, Descent, Time) ____ **38**

6.0 Conclusions ____ *39*

7.0 Issues/Future Tasks ____ *39*

References ____ *40*

Appendix A Computer Program Listings ____ *41*

1.0 Summary

The objective of the NASA Langley Advanced Air Traffic Technology (AATT) program is to develop functional designs and human-in-the-loop simulation components to provide a distributed responsibility between the flight deck and the ground-based Air Traffic Control (ATC). The specific objective of the Hazard Avoidance Planner (HAP) program is to design an airborne HAP flight crew decision aids.

The HAP computes conflict-free trajectories thirty minutes in advance and advises the crew of the impending conflicts and the resolution strategy. The HAP is integrated with the NASA Langley-developed FASTWIN aircraft, cockpit control and display simulation. It achieves a thirty-minute look-ahead by integrating equations of motion-ahead in fast time using the ADS-B information from other aircraft, and own aircraft information from the FMS. The pilot is advised of the impending conflicts and resolution strategies via information on the navigation display. The active flight plan is colored magenta, and the provisional plan generated by HAP is colored white. The pilot can activate the provisional plan, if desired. To address the problem of the uncertainty in the long-term conflict resolution strategy, the long-term probe uses the active flight plan up to a specified time and the baseline flight plan (no HAP inserted waypoints) after the specified time. The intent is to determine if the aircraft could return to the original flight-plan or perform a less severe maneuver.

A required time of arrival (RTA) function is provided for compatibility with ground-air traffic control stations which may be issuing RTA commands for traffic separation purposes, and also for separation of merging aircraft within HAP. A station-keeping function was developed to hold a desired time spacing when following another aircraft.

Electronic Flight Rules (EFR) are employed to determine the type of maneuver and the aircraft which is to maneuver (own, other, or both). The EFRs are based on concepts developed at NLR, and Eurocontrol, France. The strategies are different for head-on, crossing, and merging conflicts.

The HAP has been set up to run on a PC as an application. It can run as a standalone application with a simple own-aircraft model or with the FASTWIN FMS PC-based application.

The approach appears feasible and performs well. The look-ahead of thirty minutes provides an ample warning to the pilot and the ground controller of the impending separation problems and the proposed resolution strategy.

2.0 Problem

The conflict problem to be addressed in free flight is illustrated in Figure 2-1, which shows three air-traffic regions including unconstrained region, transition region and constrained flight region. In the unconstrained region, the aircraft are some distance from

a major airport and mainly in cruise flight. Here, conflict problems may arise from constriction due to the weather. In the transition region, the aircraft are transferring out of the unconstrained region and into the constrained region where the aircraft are following airways and being given directives by a ground-based ATC system. In the transition region, the flow continues to be more constricted which in turn may cause crossing and merging conflicts.

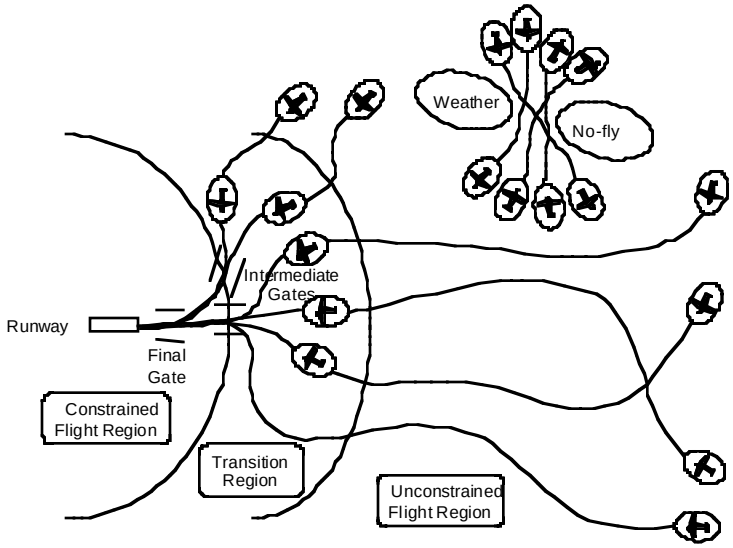


Figure 2-1. Conflict Problem

The free-flight hazard-avoidance function must provide an airborne capability to avoid these conflicts in crossing, merging and merged traffic.

3.0 Introduction

A number of approaches (see References 1-7) have been proposed for airborne conflict detection and resolution. Most, if not all, of the proposed approaches can be classified as short-term resolvers; that is, they look ahead approximately 50 miles. The concern with short-term resolvers is the air traffic controller may not have enough advanced warning of the crew's intention for resolving the conflict and may feel obligated to issue directives to avoid the conflict. Another disadvantage is the short-term planners are not designed to be compatible with the ground-based ATC system, which may be issuing a RTA command to merge and space the traffic.

Thus, the objective of this study is to develop a HAP with a thirty-minute look-ahead with RTA compatibility. Some issues to be addressed are the accuracy of the prediction at longer times, in climb and descent. The long-term prediction accuracy may not be good, because of uncertainties in winds and aircraft models. Thus, the conflict situation may be different or may not exist when the aircraft reaches the predicted conflict. In the climb and descent, accurate short-term prediction can be achieved with the current state information and constant velocity; however, the long-term prediction in climb and descent

requires intent information and more comprehensive models for the own-aircraft prediction.

4.0 Technical Approach

4.1 Concept Overview

An overview of the HAP is shown in Figure 4.1-1. A model of the own-aircraft is projected ahead in time to thirty minutes. All of the other aircraft, in the large region around the own-aircraft, are also projected ahead in time using the state information received over an ADS-B communications channel. A local solver is used to address conflicts in a local region around the projected HAP aircraft. The local solver uses conflict alert, short-term probes and conflict resolvers to identify potential and actual conflicts and resolve them. The HAP information containing the conflict problem and the resolution strategy is displayed to the crew on the navigation display.

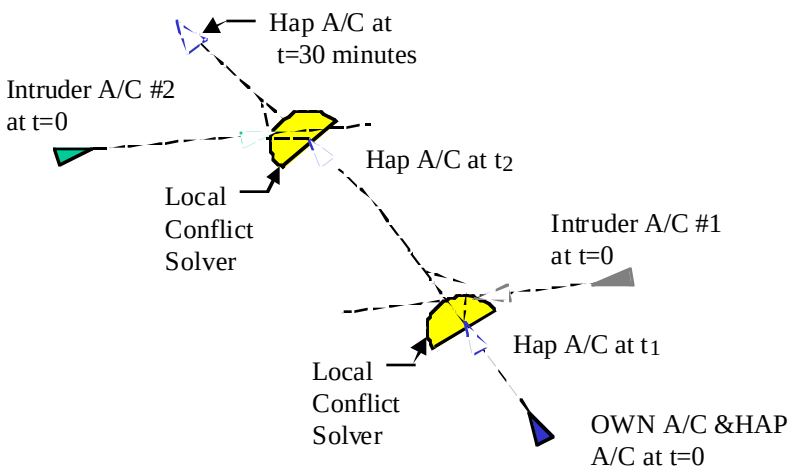


Figure 4.1-1. Concept Overview

The HAP was developed to demonstrate conflict-free aircraft trajectories using a cockpit-based approach. It is a PC-hosted application that identifies and resolves air traffic separation conflicts and provides a means to visualize and understand these conflicts. HAP is designed to run either as an integral part of the NASA Langley Research Center FASTWIN simulation, or in a standalone mode.

The HAP application consists of two separate display windows. One is a graphical window showing aircraft on a geographical map, the current flight plan of the designated (i.e., “own”) aircraft, and alternate flight plans, if conflicts have been identified and resolved. A second window shows numerical data related to the designated aircraft including current position and flight plan. Figure 4.1-2 shows the graphical display window and Figure 4.1-3 shows the data window. When running in the standalone mode, they are the only operational windows. When running with the NASA FASTWIN

simulation, three additional windows depicting cockpit displays, mode control switches, and the flight management computer are also operational. Figures 4.1-4, 4.1-5, and 4.1-6 show these windows respectively. Figure 4.1-7 shows all five of the windows on one screen.

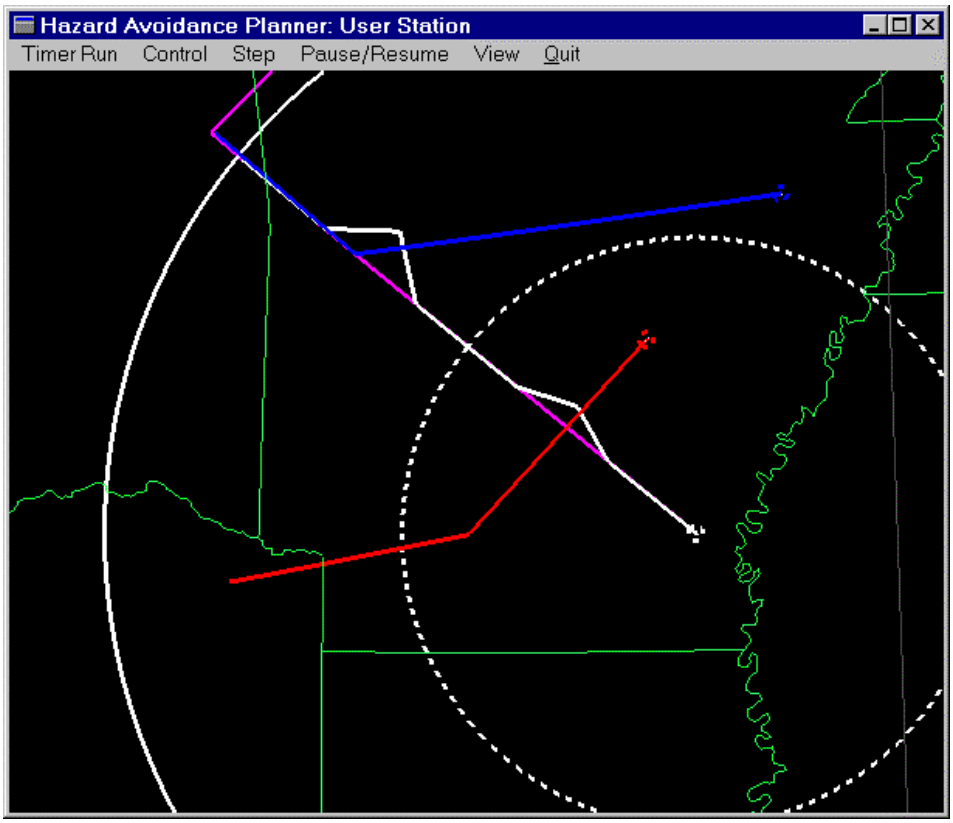


Figure 4.1-2. HAP Graphical Display Window

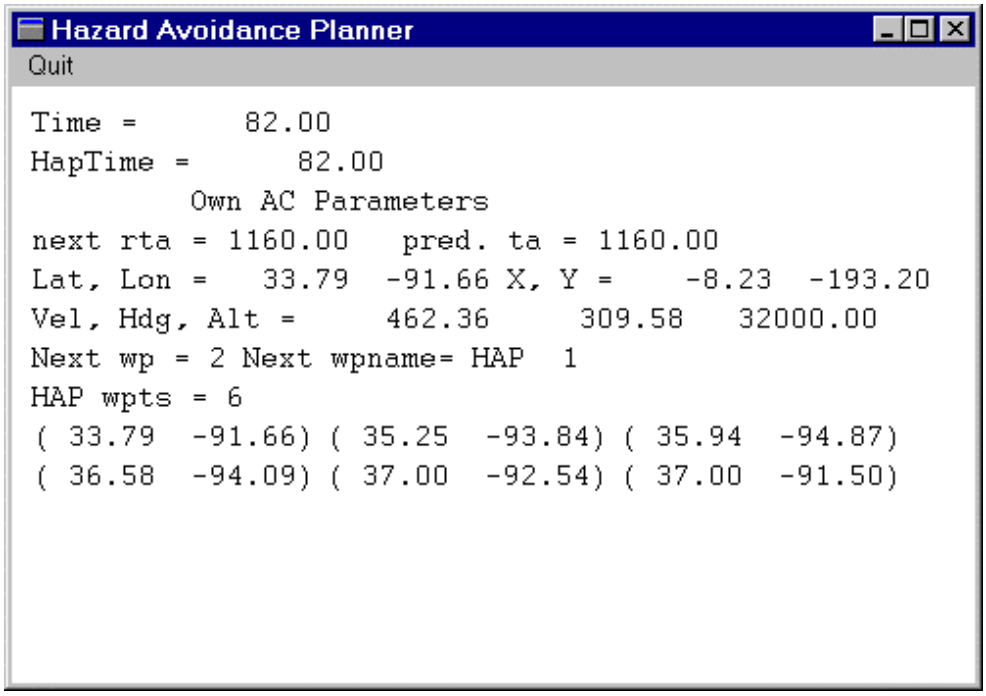


Figure 4.1-3. HAP Data Window

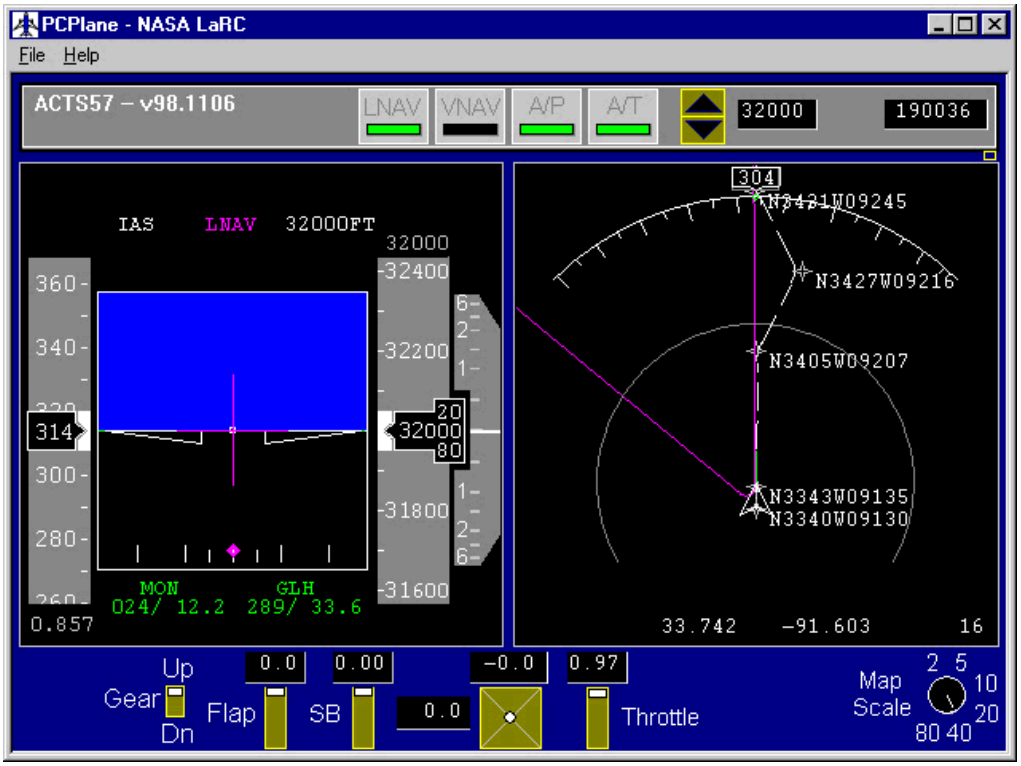


Figure 4.1-4. FASTWIN Primary Flight Display and Navigation Display

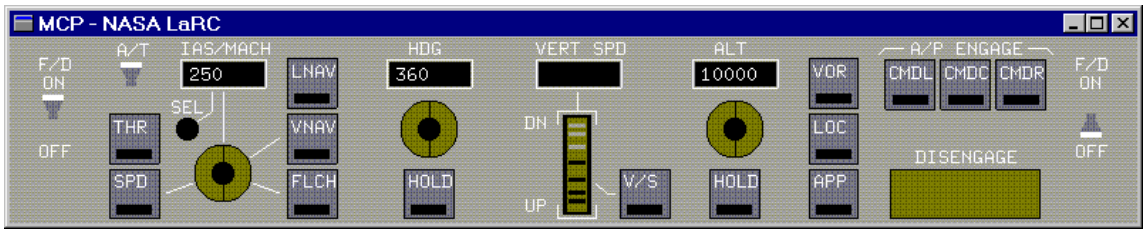


Figure 4.1-5. FASTWIN Mode Control Panel

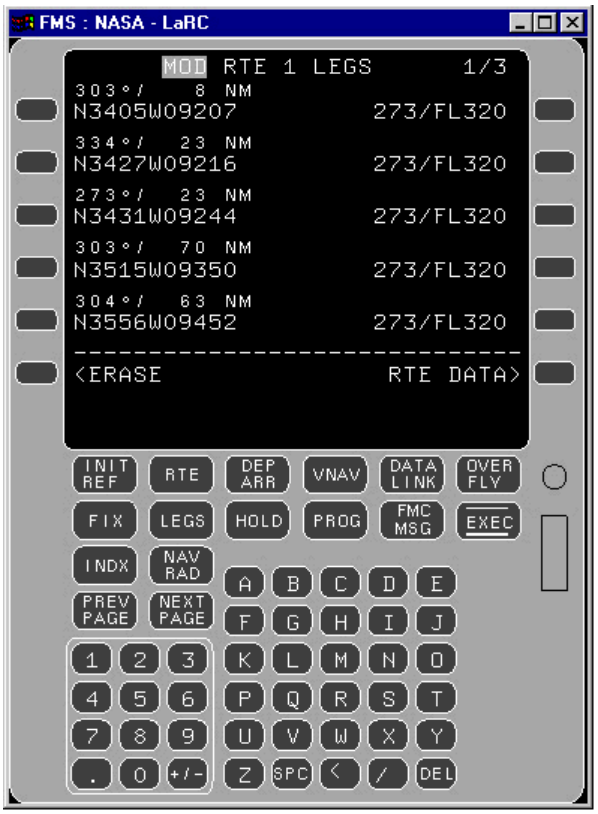


Figure 4.1.6. FASTWIN Flight Management Computer Display

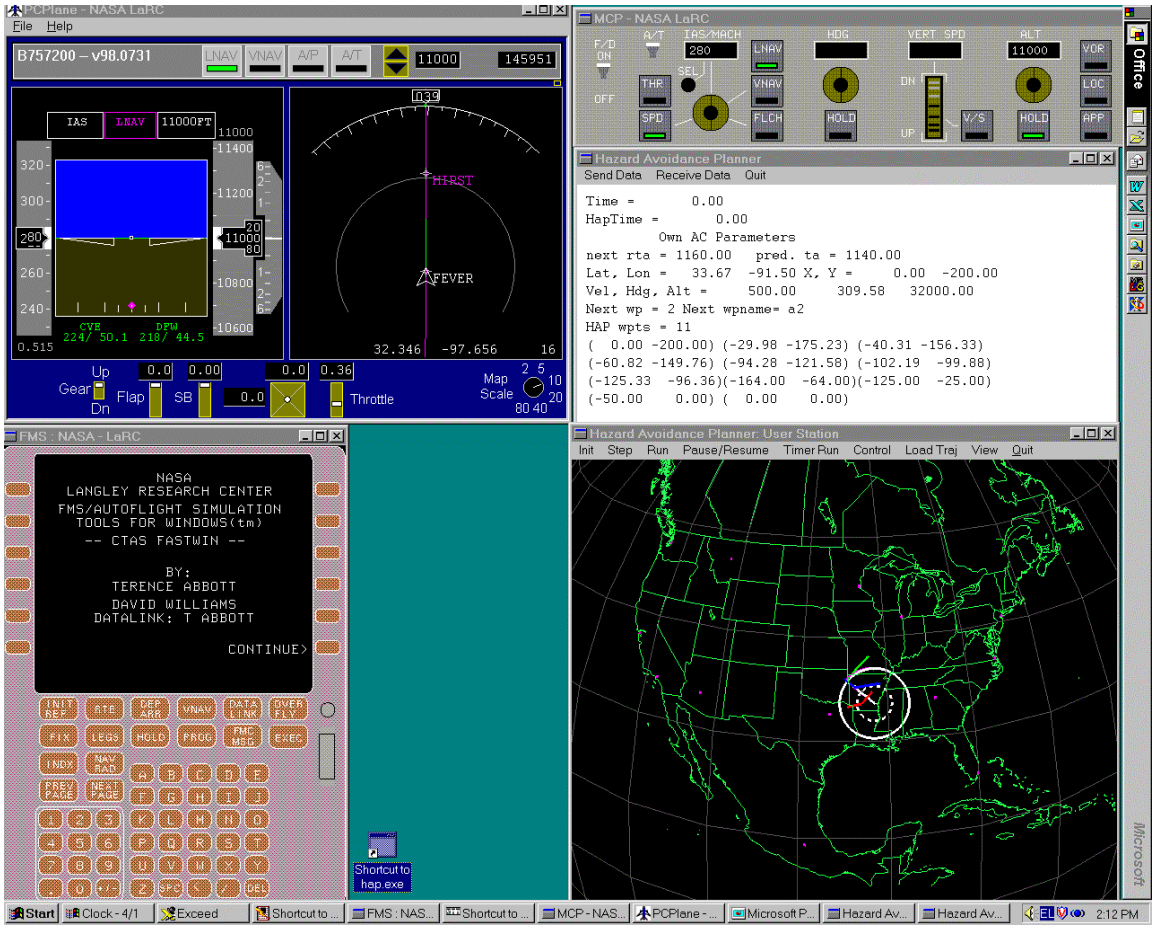


Figure 4.1-7. HAP and FASTWIN Displays

4.2 Software Overview

The software developed for the HAP program includes a simulation of multiple aircraft and multiple FMSs. It also contains the algorithms that monitor the aircraft, detect flight path conflicts and provide numerical and graphical solutions to the conflicts. This software can be divided into five categories:

1. User interface functions,
2. FASTWIN/HAP interface functions,
3. Pseudo-aircraft functions,
4. Graphics and display functions, and
5. HAP functions.

A block diagram overview of the entire simulation is shown in Figure 4.2-1 and the names of all software modules are shown in Table 4.2-1. Each of the software categories will be described in further detail using Figure 4.2-1 and Table 4.2-1 as references.

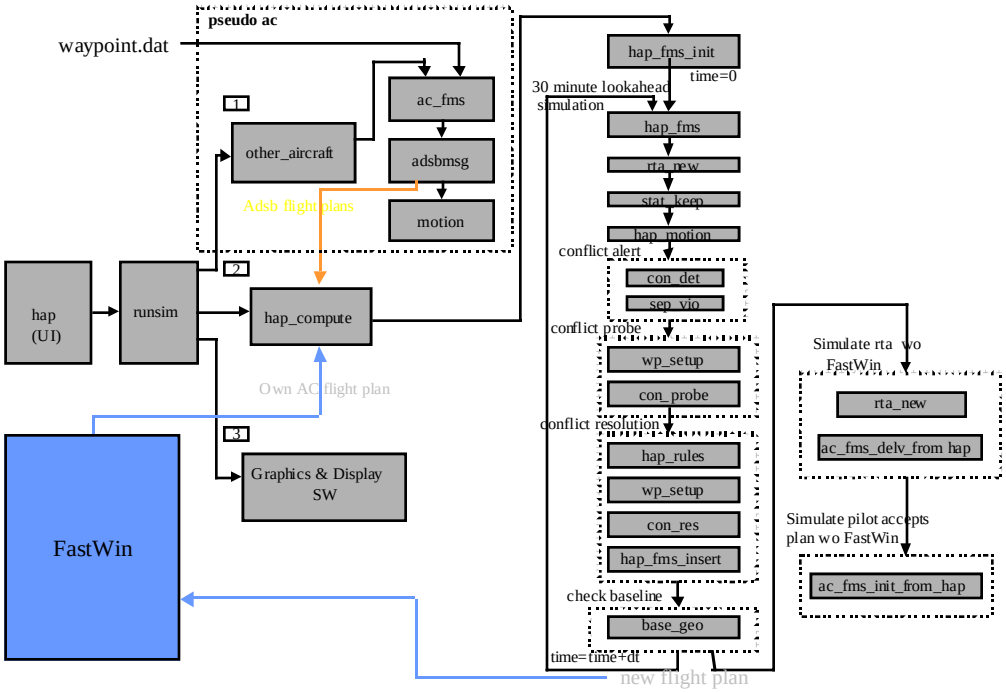


Figure 4.2-1. Block Diagram Overview of HAP Software

User Interface	FASTWIN Interface	Pseudo AC	HAP	Graphics and Display
hap.c runsim.c	fastwin_comm.c acars_utils.c F_GENI.c FW_receive.c FW_send.c GENLIB.c handleIEI.c WINNET32.c	ac_fms.f motion.f other_aircraft.f prin.f radar.f	base_geo.f con_det.f con_probe.f fortran_dim.f hap_compute.f hap_fms.f hap_motion.f hap_rules.f regroup.f rta_new.f sep_vio.f stat_keep.f wp_setup.f	airport_pcmenu.c airports.c circle.c clip.c color.c drawscene.c editaux.c font.c gauss.c gciccle.c globals.c gridio.c linked.c map.c mutils.c plane.c popfile.c protected_shape.c range_rings.c trajectory.c update_globals.c waypoints.c world.c

Table 4.2-1. HAP Application Software Modules

4.3 Software Description

A description of the HAP concept showing the time and the positional relationships between the procedures is shown in Figure 4.3-1. The HAP algorithm is based on simulating the motion of the aircraft in the system. The motion simulation is initialized using the current positions and the flight plans provided by the own-ship flight management computer and the ADS-B messages from the other aircraft. This part of the procedure is called the ***look-ahead*** or ***long-term probe.*** Each long-term probe simulates thirty minutes of flight for all aircraft and determines, at each time step, if any other aircraft is within thirty nautical miles of the own-ship. This procedure is called ***conflict alert***. If no aircraft is within the alert zone, the aircraft motion states are numerically integrated to the next time point in the probe. If other aircraft are within the alert zone, a ***short-term conflict probe*** is performed using only the conflicting aircraft. This is an additional motion simulation of the aircraft identified in the conflict alert procedure. It starts at the current look-ahead time and lasts thirty nautical miles. Separation distances between the own-ship and others are computed at each time point and compared to a threshold of eight nautical miles. If the aircraft in the short-term probe violates the eight-nautical-mile-protected zone, a lateral path ***conflict resolution*** is used to adjust flight plans until the conflict is resolved. The long-term probe now proceeds using the adjusted flight plans, or ***provisional flight plans,*** instead of the original plans. Conflict alerts, possible short-term probes and conflict resolution continue to occur, and the provisional plan has additional resolution segments added to it, when necessary. When the thirty-minute look-ahead is completed, the provisional flight plans from the simulated aircraft are transmitted back to the real aircraft where they can be displayed to the flight crews.

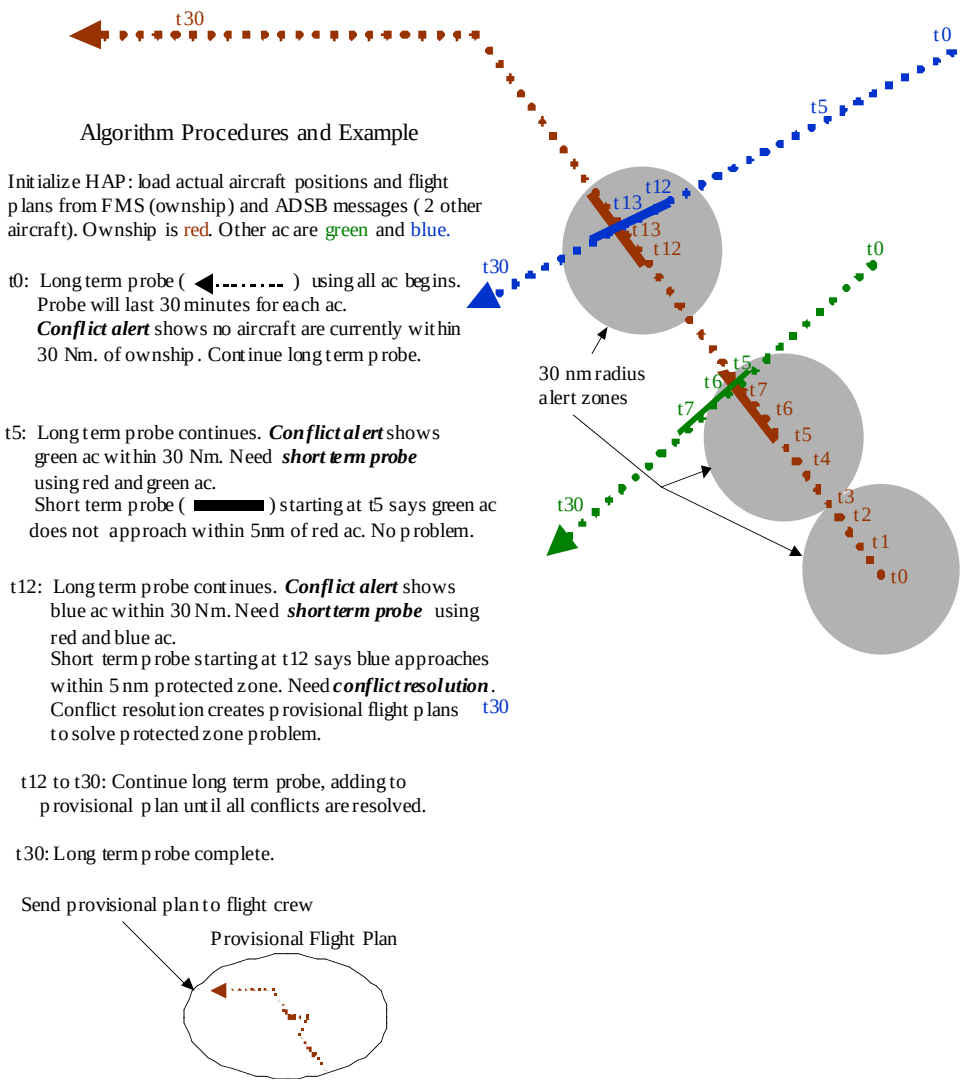


Figure 4.3-1. HAP Procedures

4.3.1 User Interface Functions

Referring to Figure 4.2-1, the boxes labeled “hap” and “runsim” are the primary user interface and executive routines. Execution and control of the application are accomplished by mouse-clicking various menu options defined in these functions.

These options include running the pseudo-aircraft simulation with and without the HAP function, selecting real-time or fast-time simulation without FASTWIN, and selecting real-time simulation with FASTWIN. Most simulation data passes through function “runsim”. This includes receiving ADSB-type messages from the pseudo-aircraft routines, receiving position and flight plans from FASTWIN, sending initial conditions to the conflict detection/resolution module, and sending conflict solutions back to FASTWIN. This routine also writes information to the data window and sends trajectory data to graphical plotting functions.

4.3.2 Pseudo-Aircraft Functions

Air traffic and ADSB message transmissions are simulated with the pseudo-aircraft functions. Subroutines, “other_aircraft”, “ac_fms”, and “motion” are the primary routines that simulate these functions. Subroutine “other_aircraft” is the executive pseudo-aircraft module. It calls the command generator, “ac_fms”, the equation of motion integrator, “motion”, and also constructs ADSB messages. The user first sets up a problem by entering aircraft flight plans into a database described in the User Procedures section of this report. Flight plans for up to 30 aircraft can be entered into a database. The maximum number of pseudo-aircraft is set at compilation time by a single parameter that is easily changed. The simple motion aircraft models use heading, velocity, and altitude commands computed from the database flight plans to determine aircraft positions as a function of time.

An ADSB-like message is then created for each aircraft. Messages contain the information needed to demonstrate the HAP software and do not have an exact ADSB-message content. Each message contains the current position, heading, and speed of each aircraft, and similar data for the “next waypoint” and “next waypoint +1”. It is assumed each aircraft provides perfect lateral position, velocity, and altitude information. Aircraft number 1 in the database is assumed to be our “own-aircraft” when operating in the standalone mode; all waypoints in the flight plan of this particular aircraft are put into the message instead of only the next two waypoints.

4.3.3 Graphics and Display Functions

Maps and data shown on the HAP graphical display window are drawn by the graphics and display functions shown on Table 4.2-1. OpenGL graphical software is used to draw:

1. The world globe with latitude/longitude lines,
2. The wire-frame geographical boundaries, and
3. The aircraft flight trajectories (time histories) created by HAP conflict detection /resolution algorithms.

Using the ADSB-message data as initial conditions, these trajectories are the result of simulating the flight paths of all aircraft in the system thirty minutes ahead in time. The look-ahead horizon is a settable parameter before compilation. If the conflict detection/resolution algorithm is not engaged, the trajectories are a time history of the aircraft positions from the start of the simulation to the current time. If the conflict detection/resolution algorithm is engaged, the trajectories show thirty minutes of simulated flight that start at the current time and use flight plans from HAP that resolve separation conflicts. Two “range rings” centered on the designated aircraft are also shown on the display; the inner one is drawn 100 nautical miles from the current position and the outer one is 200 nautical miles from the current position. Each individual aircraft in the

system also has a 2.5 nautical mile radius “protected zone” from the current simulated position, representing the desired minimum separation distance.

4.3.4 HAP Functions

The hazard avoidance functions contain the algorithms that perform:

1. Aircraft motion and integration,
2. RTA,
3. Station-keeping,
4. Conflict alert,
5. Waypoint setup,
6. Conflict probe,
7. Right of way rules,
8. Conflict resolution, and
9. Baseline geometry.

The motion functions simulate a thirty-minute look-ahead flight for our own-aircraft and all nearby-aircraft. Time of arrival functions adjust speed to reach designated waypoints at a required time, and station keeping adjusts speed to maintain spacing when one aircraft overtakes another. Conflict alert functions monitor current separation distance to nearby-aircraft during the look-ahead and waypoint setup creates an initial set of test waypoints for aircraft in a conflict group. The conflict probe functions detect future protected zone violations with aircraft identified by conflict alert. Right-of-way rules determine which conflicting aircraft are required to maneuver, and conflict resolution provides numerical solutions and provisional flight plans that will resolve these conflicts. The baseline geometry functions monitor the original flight plan to determine if previous conflicts still exist; they will return the designated flight plan to the original plan, if possible.

4.3.4.1 HAP Executive

Subroutine “hap_compute” is the executive for all the conflict detection/resolution functions. It sets up the starting conditions and calls the other functions during the thirty-minute look-ahead loop. It can also call a function that simulates a pilot accepting a provisional route.

4.3.4.2 HAP Motion

Subroutines “hap_fms” and “hap_motion” numerically integrate all aircraft positions for a thirty-minute time frame using the flight plan of our own aircraft and ADSB-message data from the pseudo-aircraft as initial conditions. “hap_fms” generates heading, velocity, and altitude commands from the flight plan messages and “hap_motion” performs the numerical integration to obtain position as a function of time.

4.3.4.3 Conflict Alert (con_alert.f)

In the conflict alert routine, illustrated in Figure 4.3.4.3-1, the distance between the HAP aircraft and all other aircraft is computed.

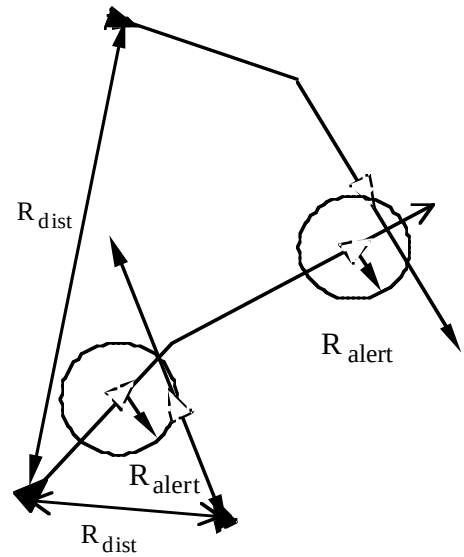


Figure 4.3.4.3-1. Conflict Alerts

The distance between aircraft (R_{dis}) is given by

$$R_{dis}(1,i) = \sqrt{[x_ac(1) - x_ac(1)]^2 + [y_ac(1) - y_ac(1)]^2}$$

If the distance is less than a separation distance, an alert is indicated. The alert logic is

$$\begin{aligned} &\text{If } (R_{dis} < R_{alert}) \text{ then} \\ &\quad \text{alert} = .\text{true}. \\ &\text{End if} \end{aligned}$$

The conflict alert geometry for a multiple aircraft encounter is shown in Figure 4.3.4.3-2. First, all aircraft within an alert radius of the HAP aircraft are identified; then all aircraft within an alert radius of these aircraft are identified.

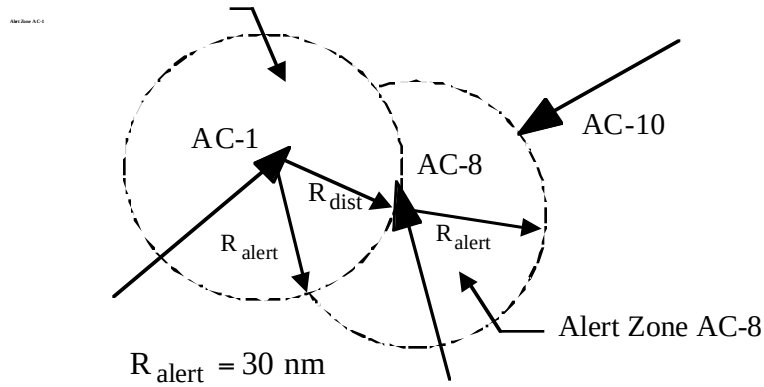


Figure 4.3.4.3-2. Conflict Alert - Multiple Aircraft

A conflict group is constructed containing all of these aircraft as shown below.

Conflict Group &
Reassigned Aircraft ID Numbers
n = igroup(j)
j 1 2 3
n 1 8 10

4.3.4.4 Waypoint Setup (WP_setup.f)

The waypoint setup routine selects the waypoints to be used either in the short-term probe or in the conflict resolution search. In the short-term probe option, shown in Figure 4.3.4.4-1, four waypoints are set up. These waypoints consist of the current aircraft position and the next three HAP aircraft waypoints, if there are four.

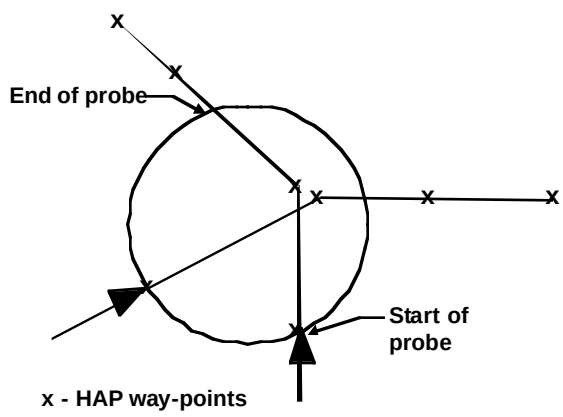


Figure 4.3.4.4-1. Waypoint Search Pattern for Short-term Probe

If the short-term probe indicates that there is a separation violation, then the waypoint setup routine is called again in the second mode. In this mode, four waypoints are set up for the conflict resolution. The four waypoints are set up using information generated

during the short-term probe. The information is the conflict entry position and headings, and the exit position and headings as shown in Figure 4.3.4.4-2.

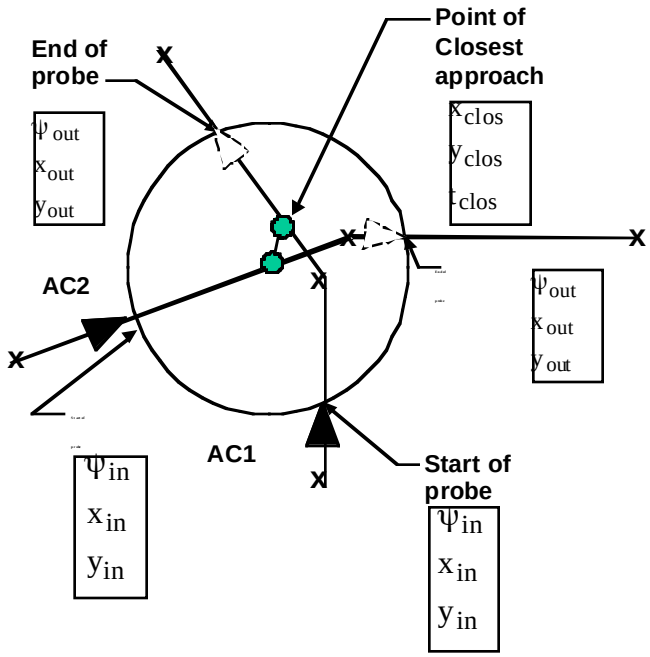


Figure 4.3.4.4-2. Conflict Probe Geometry

The waypoint setup for the conflict resolution is shown in Figure 4.3.4.4-3.

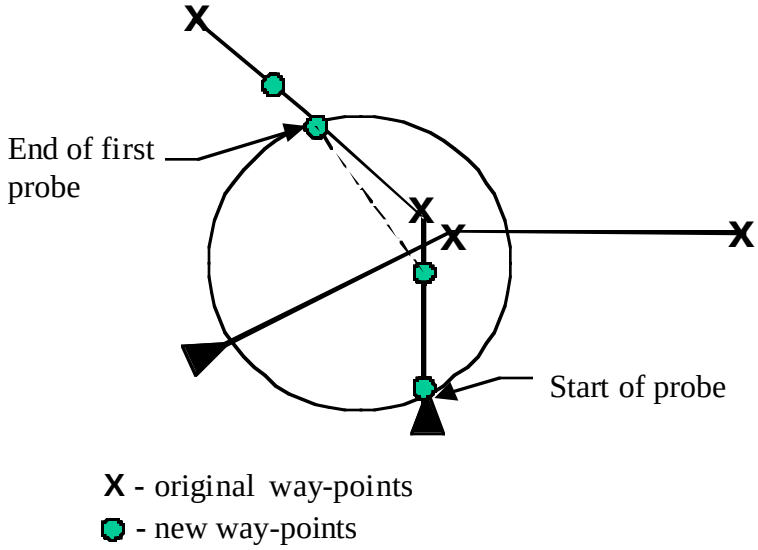


Figure 4.3.4.4-3. Waypoint Search Pattern for Conflict Resolution

The four waypoints are constructed as follows:

$$\text{Waypoint 1:}\quad wp_x(1) = x_ac_{in}$$
$$wp_y(1) = y_xc_{in}$$
$$\text{Waypoint 2:}\quad wp_x(2) = x_ac_{in} + C_1 \sin\psi_{in}$$
$$wp_y(2) = y_xc_{in} + C_1 \cos\psi_{in}$$
$$\text{Waypoint 3:}\quad wp_x(3) = x_ac_{out}$$
$$wp_y(3) = y_x_{out}$$
$$\text{Waypoint 4:}\quad wp_x(4) = x_ac_{out} + C_1 \sin\psi_{in}$$
$$wp_y(4) = y_xc_{out} + C_1 \cos\psi_{in}$$

4.3.4.5 Conflict Probe (con_probe.f)

In the short-term conflict probe, the aircraft identified in the conflict alert routine are projected ahead in time using trajectory integrators. The time of integration is limited to a preset time (typically, $t_{probe} = 200$ seconds). The lateral path is defined by four waypoints set up in waypoint setup routine. The trajectory integrators are identical to the trajectory integrators used in the motion routine. During the trajectory integration, the distance between the aircraft is computed. If this distance is less than the separation violation distance, a separation violation is indicated.

4.3.4.6 HAP Rules

If a separation conflict between two aircraft is identified, one or both of the aircraft must perform an avoidance maneuver. Criteria that determine which aircraft must maneuver are implemented in subroutine “hap_rules”. The rules of the road that are implemented in HAP are a modification of those recommended in Eurocontrol EFRs.

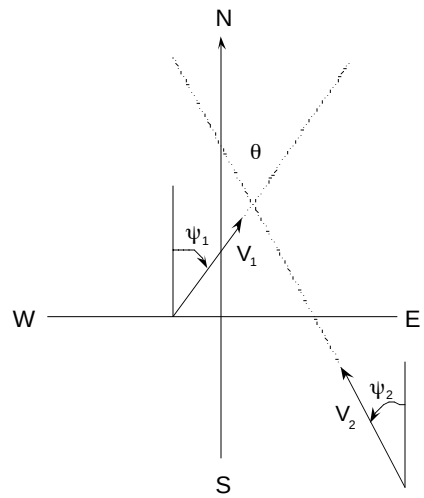
The implementation makes three assumptions:

1. All conflicts involve two aircraft,
2. Solutions will use only lateral maneuvers, and
3. Both aircraft are in a cruise flight condition.

The method used to determine the right-of-way between conflicting aircraft involves two steps. First, the type of encounter is categorized as either (1) head-on, (2) overtake, or (3) convergence, and second, the relative position of each aircraft with respect to the other is determined.

The type of encounter is determined by computing the angle of convergence between the two aircraft. Figure 4.3.4.6-1 illustrates the method and the equations used to compute the convergence angle using heading angles of the individual aircraft. Figure 4.3.4.6-2 shows the definitions of encounter types based on the convergence angle. If the convergence angle is between –165 degrees and +165 degrees the encounter is defined as

"head-on". If the angle is between –15 degrees and +15 degrees the encounter type is "overtake", and all other encounters are considered to be "convergence".

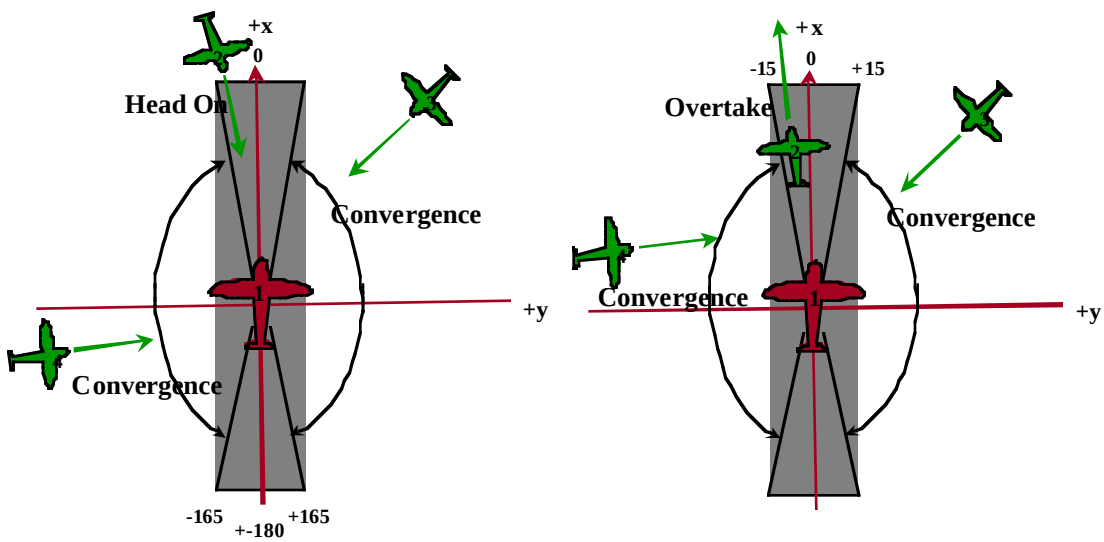


$$Vx_2 = V_2 \langle \sin\psi_2$$
$$Vy_2 = V_2 \langle \cos\psi_2$$

$$\begin{bmatrix} \tilde{V}x_2 \\ \tilde{V}y_2 \end{bmatrix} = \begin{bmatrix} \sin\psi_1 & \cos\psi_1 \\ -\cos\psi_1 & \sin\psi_1 \end{bmatrix} \begin{bmatrix} Vx_2 \\ Vy_2 \end{bmatrix}$$

$$\theta = \tan^{-1}\left(-\tilde{V}y_2 / \tilde{V}x_2\right)$$

Figure 4.3.4.6-1. Convergence Angle Definition



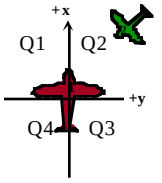
Encounter Type	Planes' Relative Direction	Angle of Convergence	Separation Distance
Head On	Opposite	- 165 to + 165 Degrees	< 5 Nm
Overtake	Same	- 15 to + 15 Degrees	< 5 Nm
Convergence	-----	Other	< 5 Nm

Figure 4.3.4.6-2. Encounter Types Based on Convergence Angle

Next, the relative positions of the two aircraft are determined. A quadrant system is used to place the other aircraft ahead of, behind, to the left, or to the right of the reference aircraft. This is done twice, using each aircraft as the reference. Figure 4.3.4.6-3 shows the quadrant system definitions and equations used to determine the relative positions.

The convergence angle and relative positions are then used to identify which aircraft must perform evasive maneuvers. If the encounter type is "head-on" both aircraft must maneuver by turning to the right. If the encounter type is "overtake" the trailing aircraft must either pass on the right or slow down and assume a station-keeping function. For "convergence" encounters, the aircraft on the right has the right-of-way. These general rules are summarized in Table 4.3.4.6-1. For the last two cases, where only one aircraft is required to maneuver, the encounter type alone doesn't provide enough information to determine the right-of-way. The relative position logic is used to determine which of the two aircraft is in trail or which is on the right. Figure 4.3.4.6-4 shows the relative position logic that is used to determine the right-of-way for these encounter types.

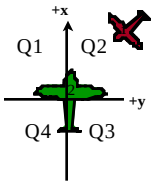
AC2 WRT AC1



$$x_{Rel1} = (x_2 - x_1)\langle \sin\psi_1 + (Y_2 - Y_1)\langle \cos\psi_1$$
$$x_{Rel1} = -(x_2 - x_1)\langle \cos\psi_1 + (Y_2 - Y_1)\langle \sin\psi_1$$

If $(x_{Rel1}\ ?\ 0\ \ \&\ \ Y_{Rel1} > 0)$ AC 2 is in Quad 1
If $(x_{Rel1}\ ?\ 0\ \ \&\ \ Y_{Rel1} \le 0)$ AC 2 is in Quad 2
If $(x_{Rel1} < 0\ \ \&\ \ Y_{Rel1} \le 0)$ AC 2 is in Quad 3
If $(x_{Rel1} < 0\ \ \&\ \ Y_{Rel1} > 0)$ AC 2 is in Quad 4

AC1 WRT AC2



$$x_{Rel2} = (x_1 - x_2)\langle \sin\psi_2 + (Y_1 - Y_2)\langle \cos\psi_2$$
$$x_{Rel2} = -(x_1 - x_2)\langle \cos\psi_2 + (Y_1 - Y_2)\langle \sin\psi_2$$

If $(x_{Rel2}\ ?\ 0\ \ \&\ \ Y_{Rel2} > 0)$ AC 1 is in Quad 1
If $(x_{Rel2}\ ?\ 0\ \ \&\ \ Y_{Rel2} \le 0)$ AC 1 is in Quad 2
If $(x_{Rel2} < 0\ \ \&\ \ Y_{Rel2} \le 0)$ AC 1 is in Quad 3
If $(x_{Rel2} < 0\ \ \&\ \ Y_{Rel2} > 0)$ AC 1 is in Quad 4

Figure 4.3.4.6-3. Relative Positions of Two Aircraft Using Quadrant System

Encounter Type	Who Maneuvers	Maneuver Rule
Head On	Both	Both Turn Right
Overtake	Trailing AC	Pass on Right or Left of Lead or Slow Down (Station-Keep)
Convergence	AC on the Left	Pass behind AC on Right

Table 4.3.4.6-1. General Maneuver Rules

Encounter Type	Location of Othership in Ownship Quadrant System (i_quad)		Location of Othership in Ownship Quadrant System (i_quad)	Who Maneuvers	Why
Head On	--		--	Both	
Overtake	3	and	1	Othership	Is overtaking
	4	and	2	Othership	Is overtaking
	1	and	3	Ownship	Is overtaking
	2	and	4	Ownship	Is overtaking
Convergence	1	and	2	Othership	Is on Left
	2	and	1	Ownship	Is on Left

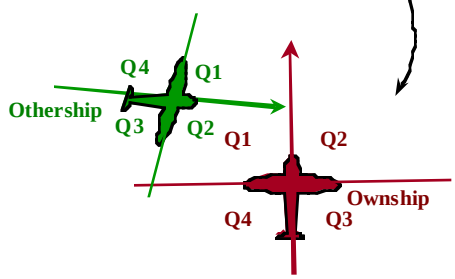


Figure 4.3.4.6-4. Encounter Type and Relative Position Used to Determine Maneuvers

4.3.4.7 Conflict Resolution (con_res.f)

If a separation violation is found during the short-term conflict probe, and the HAP rules routine determines that the conflict is to be resolved by lateral path change, then the conflict resolution routine is called. This conflict resolution routine contains logic for lateral path change to resolve the conflict. In this routine, the four waypoints set up in the waypoint setup routine are used as a starting point. Then, the aircraft motion is integrated ahead in time using the same type of motion integrator as used in the HAP motion and the conflict probe routines. If there is a conflict, the second waypoint in the four-waypoint pattern is moved as shown in Figure 4.3.4.7-1, until a conflict-free trajectory is found. Seven trajectory options are employed, one on the nominal, three to the left, and three to the right. The trajectory selected is the one that has the least deviation from the original plan and does not have a separation violation.

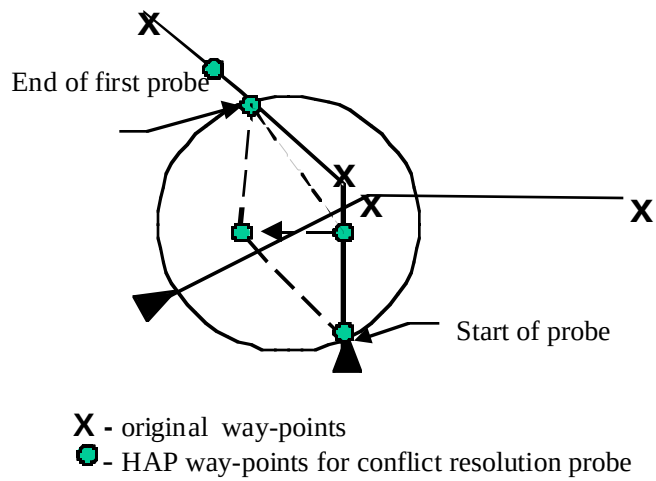


Figure 4.3.4.7-1. Conflict Resolution - Trajectory Search Approach

The geometry for the movement of the second waypoint is for the own-aircraft is shown in Figure 4.3.4.7-2.

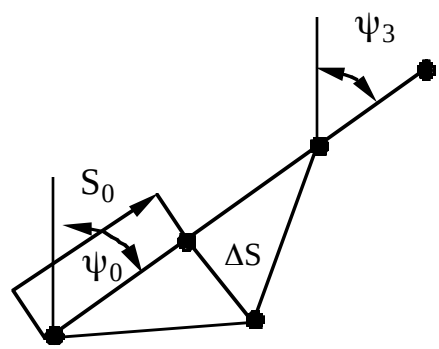


Figure 4.3.4.7-2. Geometry for Moving the Second Waypoint

The algorithm for moving the second waypoint is:

$$\Delta S = \Delta S_0 ch_dis[WP(1)]$$
$$x_wp(1,2) = x_wp(1,1) + S_0 \sin(\psi_0) + \Delta S \cos(\psi_3)$$
$$y_wp(1,2) = y_wp(1,1) + S_0 \cos(\psi_0) + \Delta S \sin(\psi_3)$$

$$\Delta S_0 = 8\,\text{nm}$$

The approach for multiple aircraft is shown in Figure 4.3.4.7-3. The maneuvers selected are the ones, which minimize the total deviation from the original plan and do not violate

the separation violation constraint. The total deviation (dev) is the sum of the deviation distances for each aircraft.

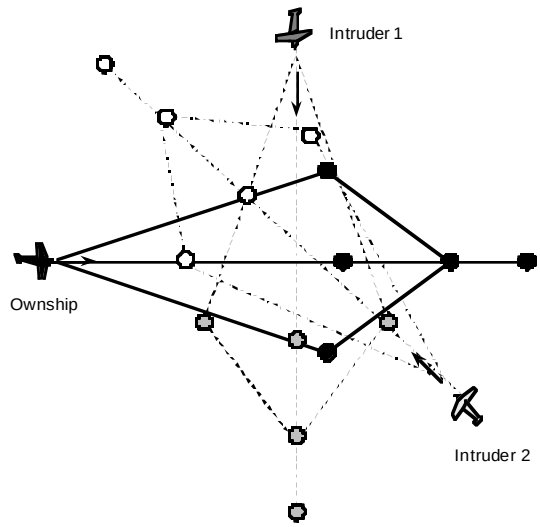


Figure 4.3.4.7-3. Conflict Resolution Search Option for Three Aircraft

The number of possible trajectory combinations is given by

$$count = ndev^{nac}$$

$$ndev = \text{Number of Trajectory Option for Each Aircraft}$$
$$nac = \text{Number of Aircraft}$$

The parameters and the search pattern for a two-aircraft cooperative maneuver is shown below. First WP(1) and ch_dis[WP(1)] are defined. WP(i) is the deviation number of each aircraft.

Type $WP(i)$	**Change Distance** $ch_dis(i)$	**Amount of Deviation** $\Delta S = ch_dis(i)\Delta S_0$	**Deviation Distance** $dev_dis(i)$
1	0	0	0
2	1	$+1\Delta S_0$	1
3	-1	$-1\Delta S_0$	1
4	$+2$	$+2\Delta S_0$	2
5	-2	$-2\Delta S_0$	2
6	$+3$	$+3\Delta S_0$	3

7	– 3	$-3\Delta S_0$	3

The search pattern is

SEARCH PATTERN
Two Aircraft - Cooperative Maneuver

Count	AC(1) WP(1)	AC(2) WP(2)	chg_dis(1)	chg_dis(2)	dev
1	1	1	0	0	0
2	2	1	1	0	1
3	3	1	– 1	0	1
4	4	1	2	0	2
5	5	1	– 2	0	2
6	6	1	3	0	3
7	7	1	– 3	0	3
8	1	2	0	1	1
9	2	2	1	1	2
10	3	2	– 1	1	2
.	.	.	.	.	
.	.	.	.	.	
.	.	.	.	.	
48	6	7	3	– 3	6
49	7	7	– 3	– 3	6

Another candidate for conflict resolution is the method used by NLR[2] shown in Figure 4.3.4.7-4. It employs an analytical method for calculating the heading change and the next waypoint; thus, it may have less computation than the search method. Some aspects of concern are that the aircraft planned their turns in the middle of the conflict, and the extension to three aircraft encounters.

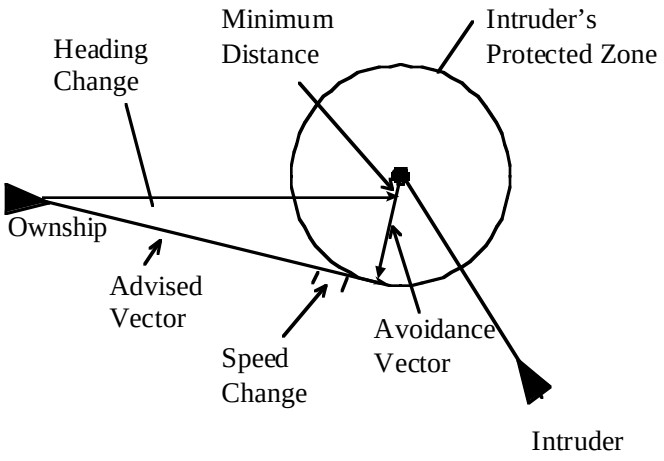


Figure 4.3.4.7-4. NLR Conflict Resolution Approach

4.3.4.8 HAP Required Time of Arrival (RTA) (RTA_new.f)

The RTA function is provided for compatibility with ground-air traffic control station, which may be issuing RTA commands for traffic separation purposes. Figure 4.3.4.8-1 shows an example of this. The RTA could be generated in HAP for separation of merging aircraft within HAP.

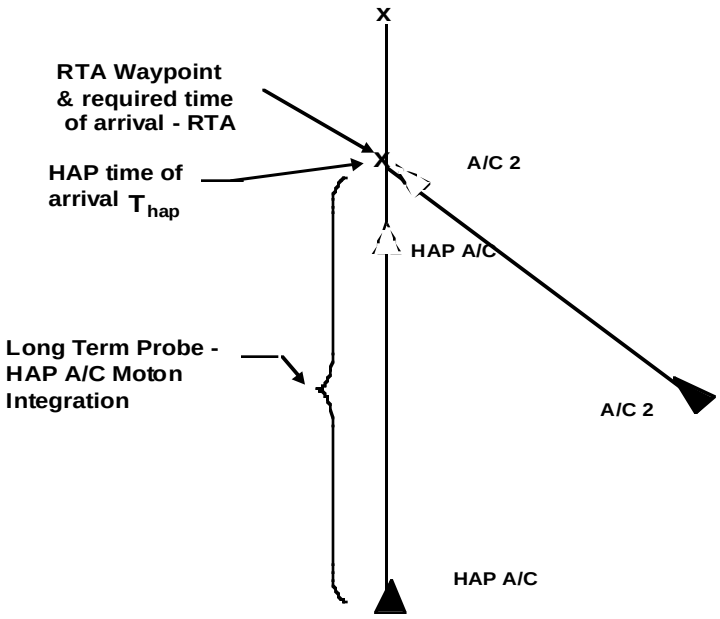


Figure 4.3.4.8-1. RTA Function

The RTA function is called from two places during a HAP long-term probe, once at the end of the run and continuously during the run.

If the HAP aircraft reaches its point of the closest approach to the RTA waypoint, the point of the closest approach is used. The point of the closest approach is used because the HAP conflict resolver may remove the external RTA waypoints from the flight plan. The logic is

$$\begin{aligned} &\text{Rdis} = \text{If } (\text{Rdis is a min}) \text{ then} \\ &\qquad T_{hap} = \text{Time} \\ &\text{End if} \end{aligned}$$

At the end of the run, the time of arrival error is computed.

$$\begin{aligned} &\text{If } (\text{Rdis is a min}) \text{ then} \\ &\qquad \Delta T = T_{hap} - \text{RTA} \\ &\text{End if} \end{aligned}$$

Then, a velocity increment to remove the time of arrival error is computed according to

$$\Delta V = K \Delta T$$

Also, during the run, the velocity increment is added to the HAP aircraft velocity and the speed limits from the aircraft FMS are applied

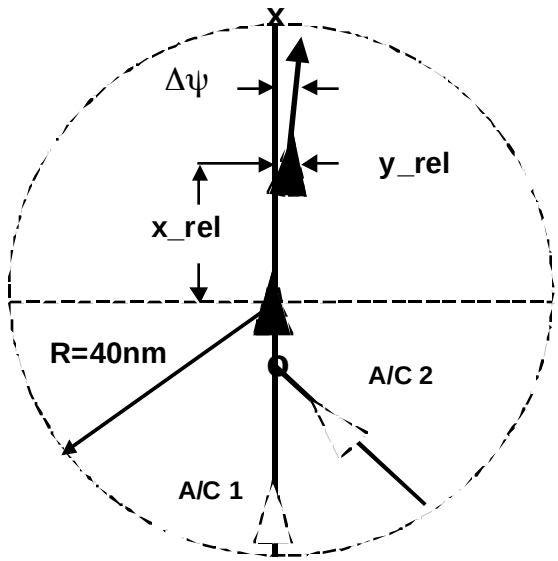
$$\begin{aligned} &V = V + \Delta V \\ &\text{If } (V < V\max) V = V\max \\ &\text{If } (V. > V\min) V = V\min \end{aligned}$$

In the standalone mode, the velocity correction term is sent to the real aircraft and added to the FMS-commanded speed.

4.3.4.9 Station Keeping (stat_keep.f)

A station-keeping function was developed to hold a desired time spacing when following another aircraft. Station keeping is performed for the HAP aircraft and the real aircraft (in the standalone mode). If the aircraft is within 40 nm, in front and on the same path and heading, then it engages in station keeping.

$$\begin{aligned} &\text{If } \{(0 < x_rel < 40\text{nm}) \text{ and } (|y_rel| < 3\text{nm}) \text{ and } (Dy < 3^{o})\} \text{ then} \\ &\qquad \\ &\text{End if} \end{aligned}$$



The engagement logic should be moved to the HAP rules.

4.3.4.10 Baseline Geometry (base_geo.f)

One problem with the look-ahead of thirty minutes is that the conflict problem, which was resolved a long time in the future, may no longer exist. Figure 4.3.4.10-1 shows an example of this. The approach for addressing this problem is to conduct the long-term probe using the active flight plan up to a specified time and then switch to use the baseline flight plan (no HAP inserted waypoints) after that time, but only if the HAP aircraft is on the baseline flight plan.

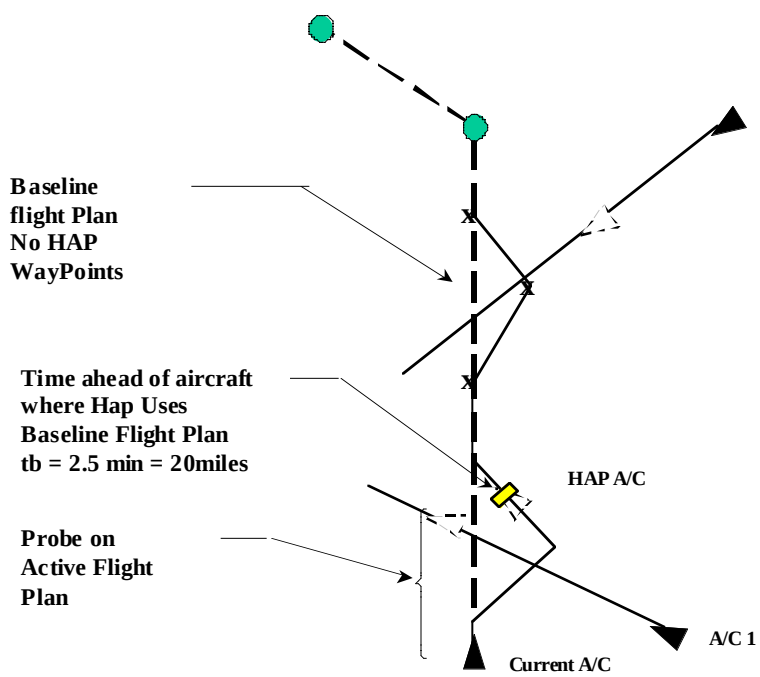


Figure 4.3.4.10-1. Future Conflict Problem

The baseline geometry logic is shown in Figure 4.3.4.10-2. The logic determines if the HAP aircraft is on the baseline flight plan and what is the next waypoint.

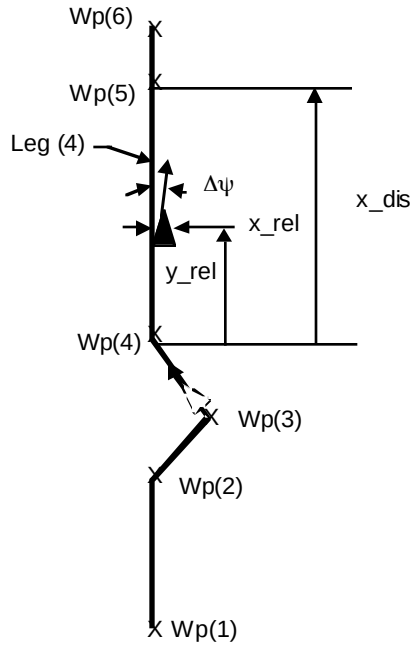


Figure 4.3.4.10-2. Baseline Geometry Illustration

Check the HAP position and heading against all waypoint legs.

If $\{(0 + 3\text{nm} < \text{x_rel} < \text{x_dis} - 3\text{nm}) \text{ and } (|\text{y_rel}| < 3\text{nm}) \text{ and } (\Delta\psi < 3^{\circ})\}$ then

On baseline flight plan

$\text{wp_next}(1) = 5$

End if

If (on baseline flight plan and HAP_time > t_b) then

Use baseline waypoints for HAP waypoints for the remaining probe

End if

4.3.4.11 FASTWIN/HAP Interface

When FASTWIN is running with HAP, the applications send and receive data using the FASTWIN interface functions. The hazard avoidance planner receives FASTWIN trajectory intent, aircraft state information, and weather information currently limited to winds aloft. These data are transmitted to FASTWIN in the form of ADSB messages formatted for compatibility with the CTAS/FMS protocol drafted for the NASA Langley/Ames TAP experiment. Aircraft-state is transmitted at 10-second intervals, while trajectory intent is transmitted upon execution of a new flight plan. However, the

updated flight plan isn't read by HAP until a new aircraft state message is received. After HAP Processing, the provisional flight plan is forwarded to FASTWIN via the same CTAS protocol mentioned above. The message is delivered to the operator in the form of a datalink message that is automatically loaded into the FMS as a modified flight plan (see Figure 4.3.4.11-1).

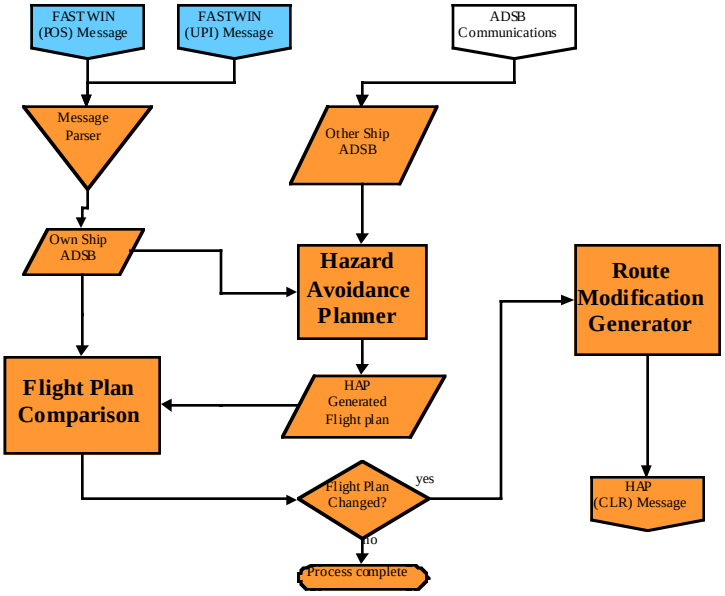


Figure 4.3.4.11-1. HAP Communication Diagram

4.4 Simulation - User Procedures

4.4.1 User Overview

The HAP is a PC-based application that identifies and resolves air-traffic separation conflicts. The primary operator interfaces are a textual database file that is prepared before running the application and the menu bar on the graphical display user station window. The HAP application is designed to run either in a standalone mode or as an integral part of the NASA Langley Research Center FASTWIN simulation. When running in the standalone mode, two separate display windows are operational. One is a graphical window showing aircraft on a geographical map, the current flight plan of the designated (i.e., "own") aircraft, and alternate flight plans if conflicts have been identified and resolved. A second window shows numerical data related to the designated aircraft including current position and flight plan. When running with FASTWIN, three additional FASTWIN applications are running with their associated windows visible.

4.4.2 Development Platform, Hardware and Additional Software Requirements

The HAP was developed using a Windows NT operating system (Version 4.0, service pack 3) and Microsoft Visual Studio 97. The Visual Studio contains C, C++, and an

optional FORTRAN compiler. The HAP application has been tested on several Dell PCs with 400 to 450 MHz processors, and also on a Toshiba laptop PC with a 150 MHz processor, all running Windows NT.

4.4.3 Compilation of HAP Application

Software modules are written using programming languages “C”, and FORTRAN. In general, the user interface and graphical displays were written in “C”, and the hazard avoidance algorithms were written in FORTRAN. Microsoft Visual Studio 97 was used as the development toolkit for the HAP application. Two separate “makefile” projects were used to compile the software modules. One project compiles all “C” and FORTRAN source; the other compiles only the “C” source files and links in previously compiled FORTRAN objects. The latter project allowed development of the user interface and graphical displays on PCs that did not have separate FORTRAN licenses.

4.4.4 Program Options Set at Compile Time

Several useful program options are currently set in the source code and may be changed by resetting a parameter and recompiling the software. Table 4.4-1 shows several of these options.

Parameter Assignment Statement	Current Value	Location	Description
#define WINTIMER1_INTERVAL	2000	hap.c (~ line 28)	Standalone mode pseudo-ac integration interval (milliseconds)
param->end	4	runsim.c (~ line 227)	Number of aircraft being simulated. Note: Database must contain at least this many (or more)
param->HapCallDt	2.0	runsim.c (~ line 239)	Interval to call hazard avoidance algorithms (seconds)
param->tmax	1200	runsim.c (~ line 251)	HAP look-ahead time (seconds)
param->lat0	37	runsim.c (~ line 252)	Reference latitude (Y_LOC=0 in waypoint.dat translates to lat0)
param->lon0	- 91.5	runsim.c (~ line 253)	Reference longitude (X_LOC=0 in waypoint.dat translates to lon0)

Table 4.4-1. Compile Time Program Options

4.4.5 Database

The HAP input database consists of 4 files, “waypoint.dat”, “terrain.map”, “airports.dat”, and “debugflag.dat”. Only “waypoint.dat” needs to be modified to set up problems and scenarios.

Database file “waypoint.dat” is a text file containing flight plans for all aircraft used in the problem. An example of “waypoint.dat” describing flight plans for three aircraft is shown in Table 4.4-2. On the first line, each aircraft is given a unique index, followed by the flight ID and the destination. The next line contains the number of waypoints followed by a separate line for each waypoint containing X and Y coordinates (nautical miles from reference point), altitude (feet), velocity (knots), and the waypoint name. This sequence is then repeated for each additional aircraft in the system.

No.	Parameters					Description					
1	**UN33b**	**MSP**				**Index, Flight_ID, Destination**					
	6					**Number of Waypoints**					
	0	-200	32000	500	a1	WP1:	X_LOC	Y_LOC	ALT	VEL	WP_NAME
	-115	-105	32000	500	a2	WP2:	X_LOC	Y_LOC	ALT	VEL	WP_NAME
	-164	-64	32000	500	a3	WP3:	X_LOC	Y_LOC	ALT	VEL	WP_NAME
	-125	-25	32000	500	a4	WP4:	X_LOC	Y_LOC	ALT	VEL	WP_NAME
	-50	0	32000	500	a5	WP5:	X_LOC	Y_LOC	ALT	VEL	WP_NAME
	0	0	5000	500	a6	WP6:	X_LOC	Y_LOC	ALT	VEL	WP_NAME
2	**AA54**	**DET**				**Index, Flight_ID, Destination**					
	4					**Number of Waypoints**					
	-80	-183.3	32000	400	b1	WP1:	X_LOC	Y_LOC	ALT	VEL	WP_NAME
	0	-141	32000	400	b2	WP2:	X_LOC	Y_LOC	ALT	VEL	WP_NAME
	185	50	32000	400	b3	WP3:	X_LOC	Y_LOC	ALT	VEL	WP_NAME
	193	50	32000	400	b4	WP4:	X_LOC	Y_LOC	ALT	VEL	WP_NAME
3	**AA421**	**MSP**				**Index, Flight_ID, Destination**					
	2					**Number of Waypoints**					
	-30	70	32000	300	c1	WP1:	X_LOC	Y_LOC	ALT	VEL	WP_NAME
	-230	-130	32000	300	c2	WP2:	X_LOC	Y_LOC	ALT	VEL	WP_NAME

Table 4.4-2. Database Example: Waypoint.dat File

4.4.6 Menu Bar

The menu bar for the graphical display window is shown in Figure 4.4-1. It consists of six items on the main menu bar and several submenus. The “Timer Run” menu item contains submenus that select either real-time operation with FASTWIN, real-time standalone operation, or 10-fast-time standalone operation. The “Control” menu is used to run the simulation either with the HAP conflict detection/resolution engaged or not engaged. The “Step” menu item causes single stepping of the standalone simulation if it is in a paused state. The “Pause/Resume” and “Quit” menu items are self-explanatory. The submenus under “View” are user-viewer options that control zoom, viewpoint over the earth, and single-frame playback of trajectories when the simulation is paused. They are primarily reminders of keyboard buttons that serve the same function, but are more convenient to use.

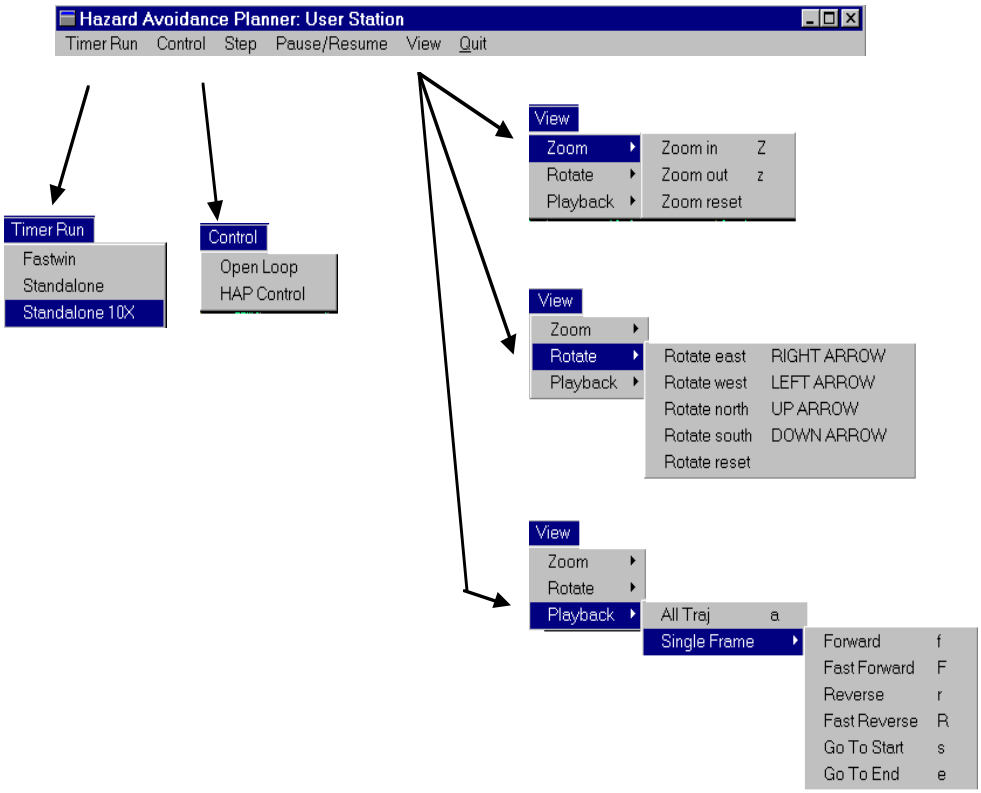


Figure 4.4-1. Menu Bar for the Graphical Display Window

4.4.7 Display Window Content

The display windows contain different information depending on the control mode.

If "Open Loop" is selected the display contains time histories of the aircraft trajectories since starting the simulation. An airplane with a 2.5 nautical mile radius protected zone around it is drawn at the aircraft's current (actual) position. This means the aircraft is drawn at the end of the trajectory. If the application is in "Pause" the "Playback" keys can be used to draw the trajectory time history data one frame at a time.

If "HAP Control" is selected the display shows trajectories generated by the hazard-avoidance algorithms. These are simulations of each aircraft starting at the current clock time and using provisional flight plans from HAP to project out to the look-ahead time (30 minutes). The time history of our own-aircraft is colored white and the remaining aircraft are assigned colors from a sequence that repeats. An airplane with a 2.5-nautical mile radius protected zone around is drawn at the aircraft's current (actual) position, like the open loop control display. Unlike the open-loop control display, this means the

airplane is drawn at the beginning of the trajectory because the trajectory represents a look-ahead from the current (actual) time. In addition to the time histories, the actual flight plan of our own-aircraft is drawn in magenta. This puts both the actual (magenta) flight plan and the provisional time history (white) of our own-aircraft on the same screen for comparison. If the application is in “Pause” the “Playback” keys can be used to draw the HAP trajectories one frame at a time.

4.4.8 Data Window Content

The data window shows the current state of our own-aircraft. It includes simulation time, the time that the conflict detection/resolution was last called, current position, velocity, heading, and altitude. It also shows the most recent provisional flight plan generated by HAP.

4.4.9 Standalone HAP

A typical sequence of events for running the HAP application in the standalone mode follows:

1. Set up a "waypoint.dat" file with desired aircraft flight scenarios.
2. Start the HAP application.
3. Resize graphic window, if desired.
4. Zoom and rotate the viewpoint to the desired region.
5. Select "Control" mode: "Open Loop" or "HAP".
6. Select "Timer Run": "Standalone" or "Standalone 10X".

4.4.10 FASTWIN/HAP

In order to run the simulation with HAP and FASTWIN communications, HAP communications must be enabled in the FASTWIN FMS. This is accomplished by editing the “F_cdufms.cfg” found in the FMS executable directory and setting the HAP communications option to “yes”. This option causes the CDU application to wait for a connection to the HAP application upon startup. After this connection is completed, FASTWIN interaction is similar to running without HAP.

After enabling HAP communications with the FMS, the following steps represent a typical start-up sequence for the simulator:

1. Set up a “waypoint.dat” file describing the trajectories of all other planes in the scenario. Note: the FASTWIN plane will override plane #1.
2. Start the HAP application.
3. Select "Timer Run": "FASTWIN" on HAP

4. Start the FASTWIN simulation suite.
5. Configure FASTWIN to run with both CDU and MCP (see FASTWIN Users Document).
6. Set initial conditions for flight on the CDU (see FASTWIN CDU Users Manual).
7. Enter flight plan into CDU.
8. Select "Run" on the FASTWIN display (right click).

As stated in step (1), the flight plan entered into the FASTWIN FMS takes the place of aircraft #1 in the “waypoint.dat” file. However, it is often useful to set up the airplane #1 for standalone running and testing the desired trajectory for the FASTWIN plane. After finalizing the trajectory, this flight plan must be manually entered into the FASTWIN FMS.

Once the FASTWIN simulation has been set up to run with HAP, and HAP is connected to the FASTWIN FMS, the FASTWIN software controls the run cycle of HAP and nothing will be displayed on HAP until data begins flowing between FASTWIN and HAP. FASTWIN provides the timing signal which causes HAP to integrate all planes and perform trajectory checking and re-planning; therefore, pausing FASTWIN will also suspend HAP processing and all pseudo-planes in the simulation. The user is still able to move the viewpoint of the HAP display while FASTWIN is in control. Additionally, the user may use the “step” function provided by HAP to check all planes’ trajectories while the FASTWIN simulation is paused.

4.4.11 Software Installation

The FASTWIN/HAP software is provided in a ZIP file that contains a directory tree similar to a standard FASTWIN distribution. A directory, named “HAP”, has been added to the main FASTWIN directory. The HAP directory contains subdirectories containing necessary data files (“DATA”), source code (“SRC”), precompiled FORTRAN objects (“OBJECTS”), and Visual C++ intermediate files (“DEBUG”). The main HAP directory also contains the Visual Studio project file and HAP executable, “hap.exe”.

Before HAP can be run, the data directory must be declared. This can be done one of two ways, 1) running HAP inside the data directory, or 2) creating a shortcut declaring the data directory as the working folder.

Running HAP inside the Data Directory

1. Copy "hap.exe" into the data folder.
2. Execute HAP.

Creating a short cut for HAP

1. In Windows Explorer, right click on "hap.exe" and select "create shortcut".
2. Right click "shortcut to hap.exe", and select "properties".
3. In the properties dialog box (Figure 4.4-2) select the "start in:" field to specify the HAP data directory.
4. Click "OK".
5. Double click the shortcut to execute HAP.

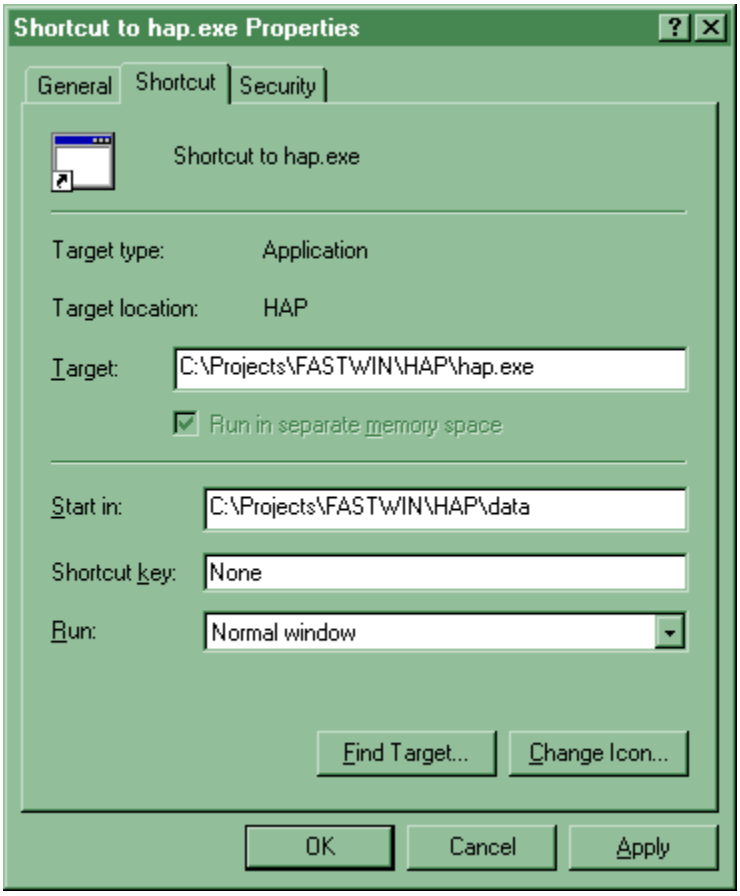


Figure 4.4-2. Shortcut Properties Dialog

After HAP has been configured to run in the data directory, the FASTWIN shortcuts must be configured to point to the correct files. Specifically, three files need to be changed: the FASTWIN batch file, the CDU link file, and the MCP link file. These files are all located in the top level FASTWIN directory.

Modifying the FASTWIN batch file

1. Right-click the “fanswin.bat” file in Windows Explorer and select “Edit”.

2. Edit the FWIN_HOME variable to indicate the directory containing the FASTWIN directory. Note: do not add the trailing "\", this is implied.
3. Save the file.

Modifying the CDU and MCP link files

1. Right-click the "mcp.lnk" file in Windows Explorer and select "Properties".
2. In the Properties dialog box (Figure 4.4-3), select the "Shortcut" tab.
3. Edit the "Target:" field. This field must contain an absolute path to the MCP executable file.
4. Edit the "Start in:" field. This field must contain the path containing the MCP executable.
5. Repeat steps 1-4 for the "cdu.lnk".

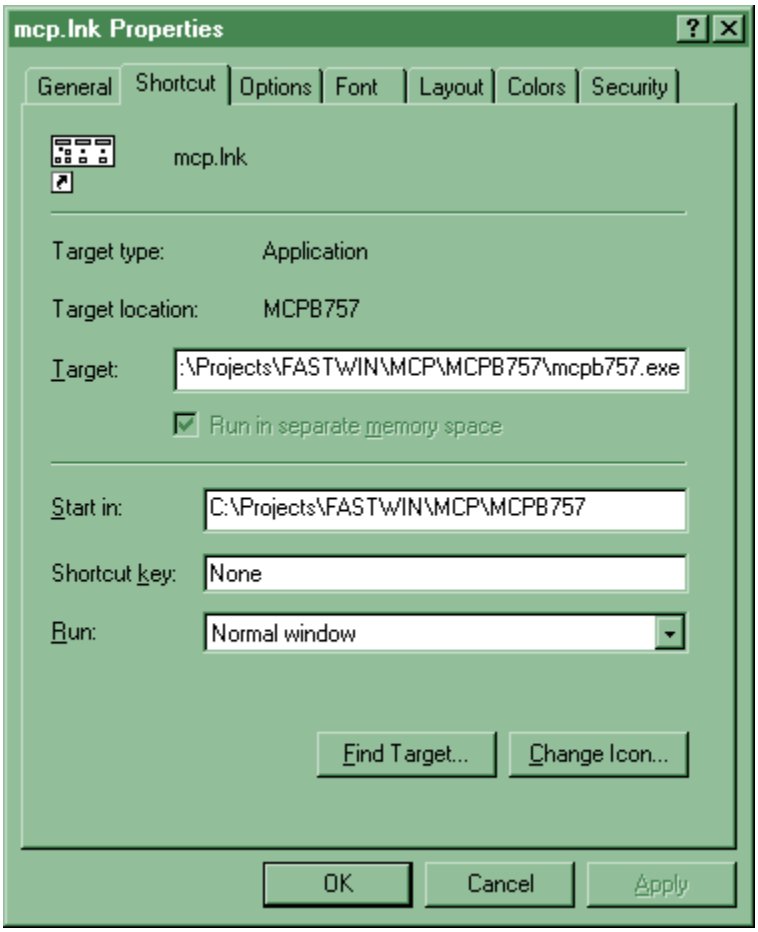


Figure 4.4-3. Link Properties Dialog

After modifying these files, FASTWIN is ready to run by double clicking the "fanswin.bat" file.

5.0 Issues and Proposed Improvements

5.1 ADS-B Issues

One problem with three ADS-B points is that during maneuvers, predictions based on three points could be incorrect. One possible method to improve the prediction accuracy is to use an estimation filter, which would retain the old ADS-B waypoints for a certain period of time as indicated in Figure 5-1.

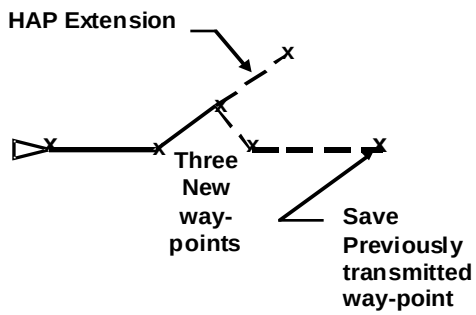
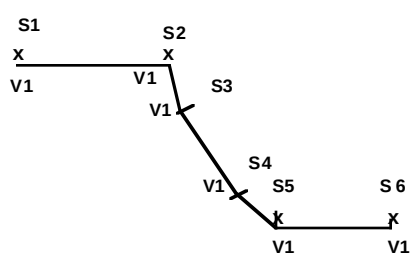


Figure 5-1. Use of Waypoint Estimation Filter

Another possible method for improvement is ADS-B message time-sharing. More waypoints information could be transmitted by transmitting in three steps. In the first message, the current waypoint, and wp+1 and wp+2 information would be transmitted. In the next message, the current waypoint, and wp+3 and wp+4 information would be transmitted. This would give the current state information every cycle and the other waypoint information every other cycle.

Another potential improvement method is to transmit, via the ADS-B, a speed profile in the prediction region as shown in Figure 5-2. This would give better prediction during climb and descent.



Velocity	Arc-length
V1	S1
V2	S2
V3	S3

Figure 5-2. ADS-B Transmission of a Speed Profile

5.2 Extension of HAP to Full 4-D (Climb, Cruise, Descent, Time)

The current HAP simulation focuses on the cruise regime and the use of speed and lateral maneuvers to resolve conflicts. The HAP motion equations are valid for cruise flight. A number of current designs for conflict detection and resolution schemes are three-dimensional. These methods, however, are short-term solvers; and thus, can use simple motion models (such as constant velocity for prediction). Prediction for thirty minutes, particularly during climb and descent, requires a general trajectory generation model. There are a number of possible approaches, e.g.

- Obtain the trajectory profile the aircraft FMS and its perturbations.
- Duplicate the trajectory generator in the FMS.
- Make the trajectory generator in the FMS accessible from the HAP.

Following is an overview description of the perturbation approach. In this approach shown in Figure 5-3, the trajectory profiles from the aircraft FMS are used for the HAP motion generation.

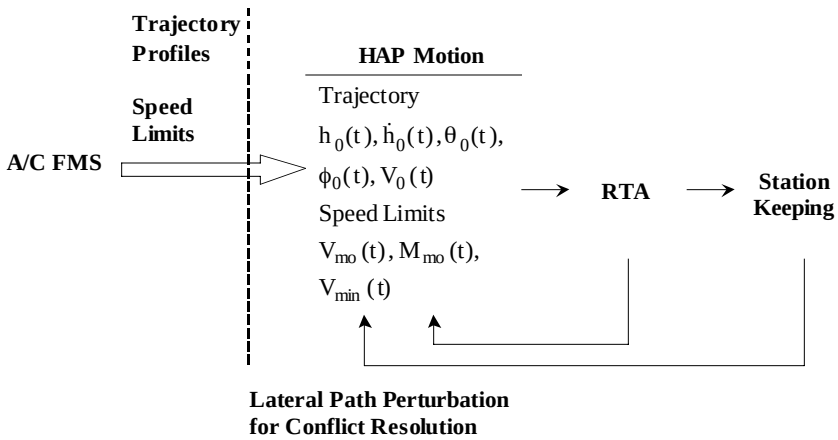
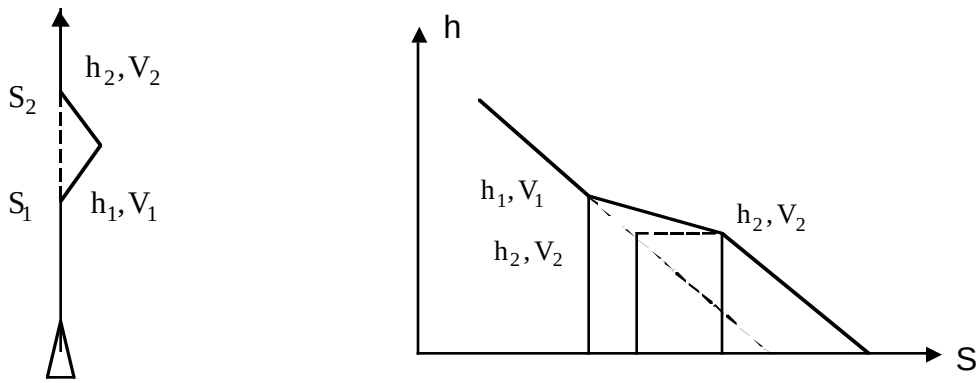


Figure 5-3. Perturbation Approach

If a conflict is detected, the path changes to resolve the conflict are handled as perturbation to the nominal path. For example, if there is a lateral path change as shown in Figure 5-4a the corresponding vertical path changes are shown in Figure 5-4b.



a) Lateral Path Perturbation

b) Vertical Trajectory Perturbation due to Lateral Path Perturbation

Figure 5-4. Perturbation Approach Illustration

All of these methods need further examination.

6.0 Conclusions

The approach appears feasible and performs well. The look-ahead of thirty minutes provides an ample warning to the pilot of impending-separation problems and to the ground controller of the future intent of the crew.

The issues of the long-term prediction uncertainty is handled adequately by using the active waypoints up to certain set-time and the baseline waypoints (no HAP inserted waypoints) after this time. This allows a change in plan as the encounter changes geometry.

7.0 Issues/Future Tasks

A number of issues remain to be addressed.

The HAP information on the navigation display regarding near-term and far-term conflicts needs further definition. The pilots need to understand when the HAP plan must be executed for conflict resolution and when HAP is only proposing a route change to return to the nominal plan, where no conflict problem may exist.

The current HAP design applies to cruise and needs to be extended to climb and descent. The current method uses the straight-line prediction, the thirty-minute look-ahead; however, will require the use of more complete climb and descent profiles from the aircraft FMS.

The HAP rules developed by NLR and Eurocontrol are for short-term conflict detection and resolution, and they need to be extended to a long-term situation. The HAP rules need also to be improved by adding RTA selection, station-keeping selection and passing

logic. The HAP simulation incorporated RTA and the station-keeping selection logic but these two were not incorporated into the HAP rules. The HAP could be improved by moving the HAP RTA and station-keeping selection logic to the HAP rules routine.

The head-on cooperative maneuver needs further examination to insure safety in the cooperative maneuver.

Because the ADS-B message contains only three waypoints, prediction problems occur when the other aircraft maneuvers. A possible improvement approach is to use a flight plan estimation filter to retain information on its flight plan.

Because the thirty-minute trajectory prediction requires significant computations, the computation of time versus accuracy needs to be continuously examined.

References

1. Kuchar, J. K. and Yang, L. C., "Survey of Conflict Detection and Resolution Modeling Methods." Paper 97-3732, AIAA Guidance, Navigation, and Control Conference, New Orleans, LA, August 11-13, 1999.

2. van Gent, R. N. H. W. et al, "Free Flight with Airborne Separation Assurance," Free Flight, 10th European Aerospace Conference, October 20-21, 1997, Amsterdam, The Netherlands.

3. Extended Flight Rules (EFR) to Apply to the Resolution of Encounters in Autonomous Airborne Separation, EATCHOP Task FCO.ETI.ST09, Version 1.0, September 1996.

4. Duong, Vu N., "Conflict Resolution Advisory for Autonomous Airborne Separation in Low-Density Airspace", Free Flight, 10th European Aerospace Conference, October 20-21, 1997, Amsterdam, The Netherlands.

5. Advanced Air Transportation Technologies: General Approach Definition of Free-Flight Conflict Prediction, NASA Ames Contract No. NASA-14290, October 30, 1996, Honeywell Technology Center, 3660 Technology Drive, Minneapolis, MN 55418.

6. Schultz, R., Shaner D., and Zhao Y., "Free Flight Concepts", Presented at the AIAA Guidance and Control Conference in New Orleans, LA, August 11-13, 1997.

7. Zhao, Y., and Shultz, R., "Deterministic Resolution of Two Aircraft Conflict in Free Flight", Presented at the AIAA Guidance and Control Conference in New Orleans, LA, August 11-15, 1997.

Appendix A Computer Program Listings