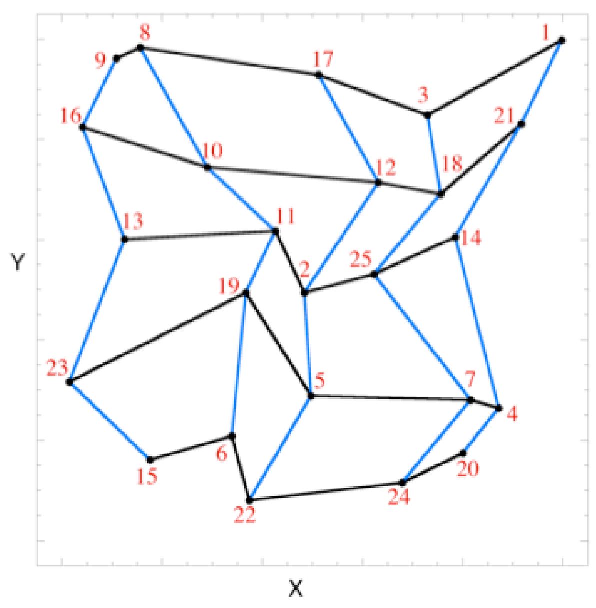
STATIONARY VERSUS MOVING GRIDS FOR AIR TRAFFIC MANAGEMENT EVALUATION

C. Kaplan, J. Dahm[§], E. Oran, N. Alexandrov*, J. Boris

Naval Research Laboratory, Washington, DC
***NASA Langley Research Center, Hampton, VA**
[§]University of Michigan, Ann Arbor, MI

The Monotonic Lagrangian Grid (MLG)

Operational Space



Computer Memory
Index Space

5	9	8	17	3	1
4	16	10	12	18	21
3	13	11	2	25	14
2	23	19	5	7	4
1	15	6	22	24	20
J/I	1	2	3	4	5

Figure 1. An example of a simple 2-D MLG, containing the x- and y-physical locations of 25 labeled nodes. The black (horizontal) lines show the x-links and the blue (vertical) lines show the y-links. The table above shows the grid indices in computer memory of the nodes shown in the figure. That is, node-15 is indexed at i =1, j = 1; node-6 is indexed at i=2, j=1, etc.

The Air Traffic Monotonic Lagrangian Grid (ATMLG) can be used to evaluate new control strategies for conflict resolution, separation assurance, and traffic flow management. The MLG is a data structure for storing the positions and other data needed to describe *N* moving objects, where *N* can be very large. The MLG sorts and orders objects on a grid structure in physical (operational) space and indexing (computer memory) space, as shown in Fig 1. Although the nodes are irregularly spaced in their physical locations (left figure), they are indexed regularly in the MLG by a monotonic mapping between the grid indices and the operational locations. Therefore, indices and physical locations of neighboring nodes are automatically known from the MLG data structure. **When using the MLG for air traffic modeling, extensive searches to find nearby aircraft are eliminated, allowing many more aircraft to be safely tracked on modest computers in real time.**

Comparison of Gridding Approaches

Grid Type	No. of neighbors checked	CPU time (s)
MLG, +/- 1 node in each direction	$3^3 - 1 = 26$	337
MLG, +/- 2 nodes in each direction	$5^3 - 1 = 124$	519
MLG, +/- 3 nodes in each direction	$7^3 - 1 = 342$	892
MLG, +/- 4 nodes in each direction	$9^3 - 1 = 728$	1481
No grid, check (N^2-1) nodes	varies; depends on number of aircraft at any one time	3584
Lat-Long Grid	varies; checks all aircraft within each static grid cell, plus in the 26 adjacent grid cells	1076

Table 1. Comparison of CPU time required for the indicated experiments.

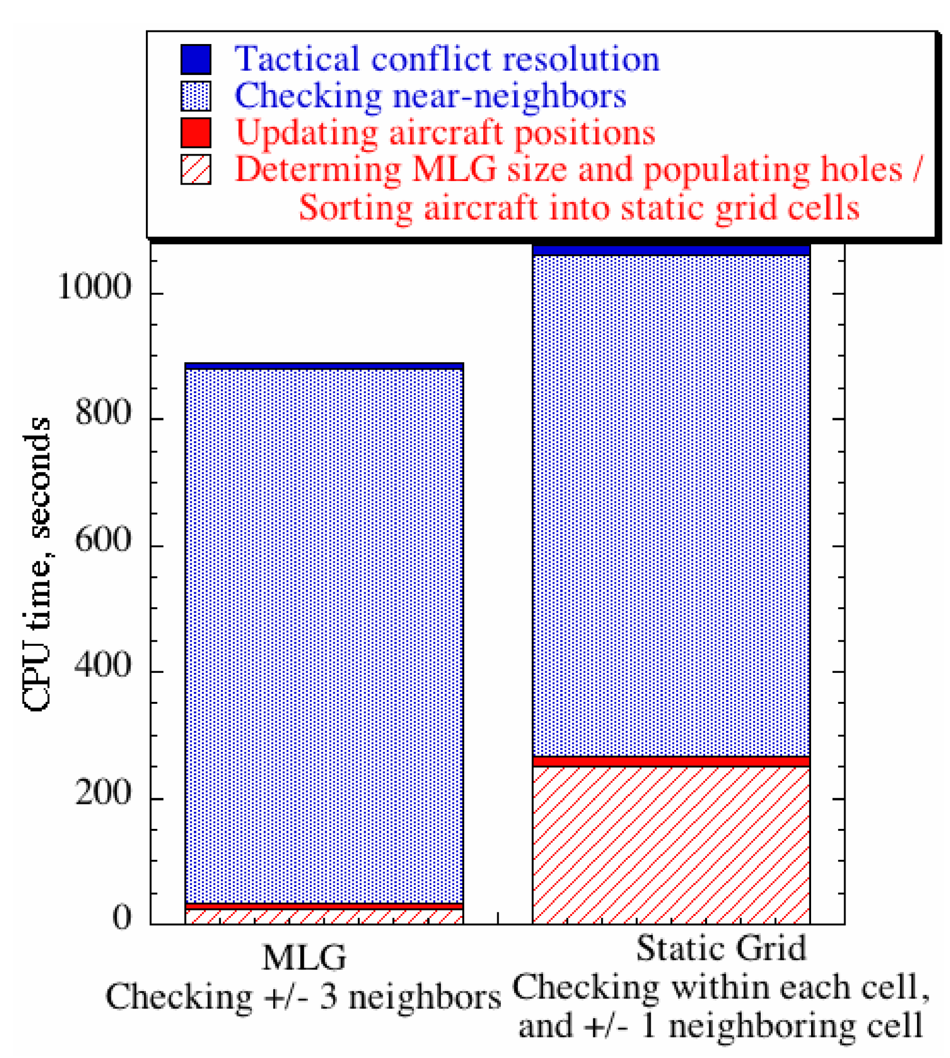


Figure 2. CPU time for MLG calculation (checking +/- 3 neighbors in each direction) and for lat-long grid calculation.

We present preliminary results from a series of numerical experiments in which we compare the use of the MLG with a static lat-long grid, to better understand the strengths and weaknesses of each approach. The calculations presented here use flight plan data from an ETMS dataset, that covers air traffic throughout the world, over a 72-hour time period in September 2006. The trajectories are calculated based on the arrival and departure airports, the intended waypoints, cruising altitude and speed of each flight. In the calculations presented below, we use the first 5000 records of this dataset, over a timespan of 14 hours of physical time. The latitude and longitude coordinates of the waypoints for each flight are converted to cartesian coordinates through a mercator projection, and the in-flight aircraft are assumed to follow straight-line paths between the waypoints specified in the ETMS file. At each timestep (equal to 10 seconds of physical time), each aircraft moves to its new position (based on velocities calculated from ETMS waypoints), and when necessary, the aircraft undergoes a tactical conflict detection and resolution (CD&R) maneuver to avoid conflicts with nearby aircraft.

Since the aircraft were positioned at locations throughout the world (based on the ETMS dataset), the lat-long grid covered a huge volume. To create such a grid, that also could be allocated into the memory of one computer processor, the static grid was composed of 134 x 125 x 8 grid cells, where each cell was of dimension 172 mi x 172 mi x 1 mi.

Table 1 shows the CPU time required for the simulations for each of the grid types. As expected, for the MLG cases, the CPU time increases as the number of nodes checked in each direction increases, and the amount of CPU time required for the non-grid case was considerably greater than for all of the other cases. The amount of CPU time required for the static lat-long grid was greater than that for the MLG when checking three nodes in each direction, and less than that when checking four nodes in each direction.

Figure 2 shows the amount of CPU time required for an MLG calculation in which three neighbors are checked in each direction (i.e., 342 nearest nodes checked per MLG node) and for the static lat-long grid. **The largest amount of time is spent checking near-neighbors.** The most significant difference between the two methods involve "book-keeping" operations: at each timestep in the MLG calculation, one needs to determine the number of nodes within the MLG (because the number of aircraft changes every timestep), whereas at each timestep in the lat-long calculation, one needs to sort aircraft into the static cells.

Discussion

When using the MLG approach, the distance between neighboring nodes varies, depending on the air traffic density at any given moment. Therefore, sometimes it may be necessary to check only one neighboring node, while at other times it may be necessary to check several neighboring nodes. In the calculations presented here, we found that 55% of the time, all aircraft within a six mile radius were only one node away, and that 30% of the time, all aircraft within six miles were only two nodes away.

To make the MLG approach more efficient, we plan to develop a method such that the number of neighbors checked varies with the air traffic density. That is, more neighbors must be checked in areas of high density, while fewer neighbors can be checked in areas of low density. Other near-term work includes exploring the use of the MLG to sort on both position and time, so that 4-dimensional aircraft trajectories can be tracked.

The authors gratefully acknowledge support from NASA Langley Research Center (LaRC) under Interagency Agreement IA1-910 between LaRC and the Laboratory for Computational Physics & Fluid Dynamics at NRL.