Published in AIAA Journal of Aerospace Computation, Info., and Comm., vol. 2, no. 9, Sep 2005

Trajectory Specification for High-Capacity Air Traffic Control

Russell A. Paielli*

NASA Ames Research Center, Moffett Field, California, 94035, USA

The doubling or tripling of airspace capacity that will be needed over the next couple of decades is likely to require four-dimensional trajectory assignment (three-dimensional position as a function of time) for appropriately equipped aircraft in high-density airspace. This paper proposes a standard trajectory specification language based on XML, the Extensible Markup Language. Trajectories are specified as a series of parametric segments. The horizontal path consists of a series of straight (greatcircle) segments connected by turns of specified radius. Altitude is specified as a low-order polynomial function of along-track position, and along-track position is specified as low-order polynomial function of time. Flight technical error tolerances in the along-track, cross-track, and vertical axes determine a bounding space, at each point in time, in which the aircraft is required to be contained. Periodic updates in the along-track axis can adjust for errors in the predicted along-track winds. Developing a consensus for an international standard is a major challenge, but a common trajectory language can greatly simplify the logistics of high-capacity air traffic control.

I. Introduction

As the demand for air transportation increases, the capacity of the current U.S. air traffic management (ATM) system will eventually be stressed to its limits. New technologies in communication, navigation, and surveillance (CNS), along with new decision support systems and an evolutionary development of the ATM system architecture,[1] can extend the capacity of the current system for several years, but a revolutionary new approach will be needed within perhaps twenty years to meet the growing demand.

An often misunderstood or overlooked fact about the current ATM system is that sector capacities are a function of controller workload rather than the airspace itself. In other words, current airspace capacity (as distinguished from *airport* capacity) is limited by the cognitive capacity of human controllers to maintain safe separation with high reliability. A controller can handle only approximately fifteen aircraft with the ultra-high reliability that is required. However, studies[2,3] have found that traffic in high-density sectors could be at least doubled or tripled over current limits without saturating the actual capacity of the airspace itself. Airspace capacity is difficult to define precisely, but it involves the rate at which conflicts arise and can be reliably resolved without causing more conflicts.

Airspace capacity could conceivably be increased by reducing sector sizes (to reduce the amount of airspace that each controller is responsible for), but that causes other problems. First, it increases the handoff workload because traffic will cross sector boundaries more often. Second, it reduces the amount of space that controllers have available to resolve conflicts within their own sector, hence more coordination is required as aircraft are diverted through adjacent sectors to resolve conflicts. The current sectorization has already reached the point of diminishing or negative return on reduction of sector sizes, so that option cannot yield the needed increases in capacity.

Because airspace capacity is currently limited by controller workload, an obvious way to increase it is to automate separation monitoring and guidance. The extreme reliability needed for such automation poses major technical challenges, however. Four-dimensional (4D) trajectories (three-dimensional position as a function of time) can at least facilitate such automation and may be indispensable to achieving it. The concept of 4D trajectories was proposed at least as far back as 1972[4] and was a key idea in the Eurocontrol

---

*Aerospace Engineer, AFC 210-10, Russ.Paielli@nasa.gov, Associate Fellow AIAA.

PHARE program,[5] for example. Several advanced ATM concepts intended for the 2025 time frame[6–10] are also based on 4D trajectories. However, a standard format for specifying continuous 4D trajectories, with error tolerances in all three axes, does not exist, nor is one currently being developed by any major standards organization.

The methods proposed in this paper are intended to support a particular concept of operations for future ATM. That ATM concept is not the subject of the paper, but it will be used as a default to provide some context for how trajectory specification could be used in practice. Note, however, that the methods proposed in this paper could apply equally well to any other ATM concept that is based on precise 4D trajectories, such as the ones cited in the previous paragraph.

The default ATM concept for this paper is referred to as the Advanced Airspace Concept (AAC).[9,10] AAC allows pilots or airlines to specify a desired trajectory and downlink it to a centralized ground system, which checks it and approves it if it is free of conflicts (and is consistent with traffic flow limits). If the requested trajectory conflicts with an existing trajectory assignment, the ground system minimally modifies it to resolve conflicts then uplinks it as an assigned trajectory. If no requested trajectory is submitted, the ground system generates a reasonably optimal one that is as consistent as possible with any specified preferences. An automated safety-critical backup system[11] provides tactical separation guidance for unequipped and non-conforming aircraft. A key characteristic of AAC is that it implements centralized *coordination* without requiring centralized *control*.

The stringent regimen implied by assigned 4D trajectories may seem to contradict the notion of "free flight," but it actually does not. The objective is not to restrict routing options any more than necessary, but rather to track intent with high reliability and precision. Rather than trying to "predict" the trajectory of each aircraft, with no guarantee of correctness or even attempted conformance, trajectories can be specified precisely, and conformance can be mandated. Without such a regimen, airspace capacity can never be safely maximized. Note, however, that pilots and airlines can be allowed to request trajectory revisions at any time (within reason), and if a requested trajectory is free of conflicts (and consistent with the traffic flow limits), it should be approved. Flight can therefore be as "free" as possible without jeopardizing safety.

A major safety benefit of the proposed regimen is that all equipped aircraft can be guaranteed to have mutually conflict-free trajectories to fly for several minutes or more, even during a complete failure of all ground systems and the entire communication infrastructure. The duration of this conflict-free period will depend on how long aircraft can maintain conformance without updates, which in turn depends mainly on wind modeling accuracy. During periods of high accuracy, the conflict-free period could be indefinitely long. This benefit could ultimately prove to be critical for the acceptance of automated separation in high-density airspace. Without 4D trajectory assignments, a ground computer or communication system failure could dump the responsibility for safe separation onto human controllers, which would be unacceptable. Recall that increasing the traffic density beyond what a human controller can reliably handle is the *objective* of automated separation.

A standard called Controller/Pilot Datalink Communication (CPDLC)[12] is currently being developed for communicating specific maneuvers using standard message types, but it is not designed to specify 4D trajectories. Barrer proposed the concept of "path objects",[13] which constitute a simple "path language" for expressing 3D route patterns such as straight segments, turns, S-turn delays, holding patterns, etc. CPDLC and path objects are potentially useful, particularly in the period of time before 4D trajectory assignment can be implemented, and for aircraft that are not equipped for 4D guidance, but they do not actually specify continuous 4D trajectories. The FAA is developing the concept of a "flight object,"[14] which will contain 4D trajectory predictions, but apparently no information has yet been published with regard to how trajectories would actually be represented.

Barhydt and Warren[15] recently proposed "The Development of an Information Structure for Reliable Communication of Airborne Intent and Aircraft Trajectory Prediction." Their proposal is associated with Automatic Dependent Surveillance-Broadcast (ADS-B).[16,17] Because it is intended for implementation within the next few years, it is constrained by current and next-generation FMS (Flight Management System) capabilities. The ADS-B Trajectory Intent Bus gives discrete 4D waypoints but does not precisely specify a 3D reference position as a continuous function of time, nor does it precisely specify a 3D bounding space at each point in time. ADS-B was designed for state dissemination rather than detailed trajectory specification, and attempting to use it for the latter would be awkward at best. Also, problems with using a sequence of discrete 4D waypoints will be discussed in the next section.

The objective of this paper is to propose a standard and a parametric format for specifying 4D aircraft trajectories, a standard 4D “path language.” The specified trajectories could be entire flights from takeoff to landing, or any portion thereof. This standard will precisely specify the assigned 3D reference position and flight technical error tolerances as a continuous function of time. At each point in time a 3D bounding space will be determined in which the aircraft is required to be contained. This bounding space is similar in principle to the PHARE “contract tube,”[18] but it will typically be larger and more flexible, particularly in the along-track direction. Trajectories can then be synthesized to guarantee the minimum required separation for any pair of aircraft as long as both conform to their assigned trajectories within the specified tolerances.

A key aspect of the format proposed in this paper is that it is based on XML, the Extensible Markup Language. XML is a text-based format that is rapidly replacing binary formats for automated business-to-business (“B2B”) transactions and is being widely used for computing standards such as Scalable Vector Graphics (SVG). Whereas binary formats typically require the same data to be transferred in the same precise order every time, XML provides more flexibility in the selection and ordering of the data fields. The flexibility of XML will be indispensable for trajectory specification because each trajectory can have a variable number of segments of various types. XML also allows aircraft characteristics and flight preferences to be easily and clearly specified, which will be discussed later in the paper. Note that real-time decision makers such as pilots and controllers will not use the XML text directly but can be provided with a graphical interface to view and modify trajectories when necessary.

The remainder of the paper is organized as follows. First, the basic requirements of the proposed trajectory specification standard are discussed. Then the necessary coordinate systems and transformations are outlined. Next, polynomial approximation of vertical profiles and along-track position is discussed. The proposed XML format itself is then presented. Finally, routine along-track trajectory updates are discussed.

II. Requirements

The trajectory specification standard to be proposed in this paper is intended to be used for communicating trajectories between aircraft and other aircraft or ground systems. Pilots or airlines should be able to use it to downlink requested trajectories, and ground systems should be able to use it to uplink assigned trajectories. The basic requirements are that it be:

- able to precisely specify any “reasonable” 4D reference trajectory.
- able to precisely specify error tolerances relative to the reference trajectory.
- based on a global earth-fixed coordinate system.
- parametric and reasonably compact.
- based on a text format readable by humans.
- suitable for an international standard.

The first requirement is that the format be able to precisely specify any “reasonable” 4D trajectory (3D position as a function of time). A unique 3D position must be precisely determined at each point in time, and the set of specifiable trajectories must not be unreasonably restrictive. Efficient climbs and descents must be allowed, for example, and turns must be allowed during climb and descent. The horizontal path will be restricted to straight (greatcircle) segments connected by turns of constant radius to simplify computations and conformance monitoring. These restrictions should not significantly limit practical routing flexibility. Note that wind-optimal routes can be approximated with sufficient accuracy for practical purposes using greatcircle segments of, say, 100 to 200 nmi in length (depending on the length of the flight). More general horizontal path segment types can be added later if desired.

The second requirement is the ability to specify error tolerances for the flight technical error in each of the three axes: along-track, cross-track, and vertical. The error tolerances relative to the reference trajectory discussed in the preceding paragraph will precisely determine a 3D bounding space in which the aircraft is required to be contained at any point in time. Those bounds will be the key to assuring that the minimum required separation is maintained at all times without the attention of a human controller. If an aircraft fails to conform, or is expected to fail shortly, its status can be temporarily downgraded to unequipped, and it can be automatically issued a basic heading or altitude resolution advisory, if necessary, but such remedies depend on the particular concept of operations and are outside the scope of this paper. Note that the term "error" is used to denote any deviation from the reference trajectory, but such errors are allowed if they are within the allotted error tolerances.

Trajectories can be synthesized to guarantee the minimum required separation for a specified period of time called the conflict time horizon, which could be perhaps fifteen minutes. The key point is that, if the trajectories are correctly synthesized, conformance by any two aircraft will guarantee the minimum required separation between them for a specified period of time, regardless of where each aircraft is within its bounding space. In other words, the bounding spaces themselves must always maintain the minimum required separation. Note that minimum separation standards are specified in terms of the separation *distance* between aircraft, regardless of velocities or higher-order dynamics. Hence, the trajectory error tolerances will also be specified in terms of distance or length. Velocity and acceleration can obviously affect future conformance, but actual current conformance will not depend on them. Nevertheless, a conformance monitoring system is free to use velocity and acceleration to try to predict impending nonconformance.

In the current air traffic system, standard navigational conformance bounds of ±4 nmi in cross-track define a lane width of 8 nmi. However, those bounds are routinely violated for various reasons, such as loose piloting or controllers issuing heading "vectors" or "direct-to" clearances (to skip flightplan waypoints) but not entering them into the Host Computer System (HCS). In the vertical axis, conformance bounds of ±200 ft apply only in level flight, and no bounds apply in the along-track axis (except arrival time constraints). The lack of rigorous conformance bounds in the current system makes conformance monitoring a "fuzzy" problem, which Reynolds and Hansman[19] have attempted to solve using fault detection methods. But conformance monitoring itself is precisely defined if conformance bounds are based on position only and specified precisely, as proposed in this paper. The more difficult and "fuzzy" problem is then the detection of faults that could lead to imminent non-conformance, which is where Reynolds' approach could still apply.

The error tolerances should be based on Required Navigation Performance (RNP) specifications,[20] and they should be set so that all equipped aircraft are capable of conforming with near certainty. The error tolerances would normally be set by ground systems based on aircraft equippage and traffic density. The tolerances could be relaxed in sparse traffic when tight tolerances are unnecessary. Aircraft equipped for tighter RNP specifications could be favored in arrival slot assignment or conflict resolution (e.g., by making the less-equipped aircraft maneuver to resolve), but such considerations are beyond the scope of this paper.

Because winds cannot be modeled or predicted exactly, the most challenging axis for which to set tolerances is the along-track axis. Tightening the along-track tolerance increases airspace capacity, but it also increases the probability that aircraft will be required to fly at inefficient or even unflyable airspeeds. Along-track position error tolerances must be set as a compromise between those two effects. For more flexibility, they can be allowed to grow linearly with time. Also, the along-track assigned position and velocity can be updated periodically to compensate for errors in modeling and prediction of along-track wind magnitudes. Such updates should only be allowed, however, when they do not cause a conflict.

The next requirement is that the format be based on a global earth-fixed coordinate system, which will provide a common reference. Local coordinate systems, such as the (pseudo-Cartesian) stereographic projection used within each Air Route Traffic Control Center (ARTCC, or "Center"), are inappropriate for enroute airspace because they are each valid only within one Center. The complexity of switching coordinate systems for each Center would be unnecessarily complicated. The standard WGS84 geodetic coordinate system (latitude, longitude, and altitude above the reference ellipsoid) will be used as the reference coordinate system for enroute airspace. Local coordinate systems might be convenient in terminal areas however, so they should be available too. Local airport coordinate systems can make the position of an arriving aircraft relative to the runway more obvious, for example. Also, a curvilinear flightpath coordinate system will be introduced in the next section for specifying and monitoring the flight technical error tolerances.

The fourth item in the requirements list is that the format be parametric and reasonably compact. A continuous 4D trajectory can be approximated by a simple sequence of discrete 4D points (t,x,y,z), but that tends to be inefficient in terms of storage and bandwidth. More importantly, it also fails to capture the structure of the trajectory. Real trajectories consist of discrete segment types, such as climb at constant CAS (Calibrated Airspeed), cruise at constant Mach, etc., but discrete 4D points do not convey that structure. Aside from making the trajectory harder for humans to comprehend, this lack of structure forces the FMS to do extra computation to determine flight modes and mode switch points. This paper will propose a structured, parametric approach based on straight (greatcircle) segments, constant-radius turn segments, and low-order polynomial approximation.

A more fundamental problem with using a sequence of discrete 4D points is that along-track position error couples into cross-track and altitude. The Suppose, for example, that an aircraft is on approach for landing and is one minute behind schedule (but still within tolerance). If altitude is specified as a function of time, the aircraft will be required to land several miles before it reaches the runway! On the other hand, if altitude is a function of along-track position, the aircraft will be required to land at the runway regardless of its status with respect to its schedule. Clearly the latter is preferable. A 4D trajectory assignment should properly be regarded as a 3D earth-fixed tube, where the position along the tube is the fourth dimension. Although discrete 4D points are good for specifying trajectories that have already been flown, they are simply not the best choice for specifying trajectories that are yet to be flown.

The fifth requirement listed above is that the format be in plain text, readable by humans. The traditional standard for computer text (ASCII, or American Standard Code for Information Interchange) is more than adequate for this application. Text-based formats typically provide less efficient storage than binary formats, but they also tend to be more flexible and less prone to error. Also, text-based formats are more convenient because they can be read directly by humans. This is certainly not to imply that the text is the best way to represent trajectories for all purposes, of course. A graphical representation is obviously preferable to text for real-time decision makers such as pilots and controllers, but text is preferable to binary data for engineers and analysts who need to examine the data in more detail off-line.

XML, the Extensible Markup Language,[21] is the new standard text-based format for specifying structured data and transferring it across heterogeneous computer platforms independently of any particular programming language. Whereas binary formats typically require the same data to be transferred in the same precise order every time, XML provides more flexibility in the selection and ordering of the data fields. The flexibility of XML will be indispensable for trajectory specification because each trajectory can have a variable number of segments of various types. The flexibility will also allow trajectories to be updated without repeating all the data that remains unchanged from the previous update, which could more than compensate for the inherent inefficiency of text-based data. XML text can also be compressed for more efficient use of bandwidth, of course.

The final requirement listed above is that the proposed trajectory specification standard be suitable for an international standard that is recognized by, and can be automatically flown by, any standard FMS. The standard could be used onboard aircraft to downlink requested trajectories constructed by the FMS or by the pilot using a graphical interface. It could also be used by ground systems to check for conflicts and to approve or uplink assigned trajectories. Developing a consensus for an international standard is obviously a major challenge, but such a common language can greatly simplify the logistics of high-capacity air traffic control. With a common trajectory language, the chances of miscommunication will be much less than they would be without one. The objective of this paper is to highlight the need for such a language and to suggest a possible starting point. If adopted, the actual communication mechanism would probably be an extension of CPDLC[12] or a new datalink message over the Aeronautical Telecommunication Network (ATN).

III. Coordinate Systems and Transformations

For the purposes of this paper, a trajectory specification consists of a 4D reference trajectory and associated flight technical error tolerances. The reference trajectory is the precise 4D trajectory the aircraft would fly in the ideal case of zero flight technical error. It is a precise 3D position that varies as a function of time, and the position at any point in time will be referred to as the reference position. The error tolerances, on the other hand, are the maximum allowed error in each of the three axes: along-track, cross-track, and vertical. These tolerances define a 3D bounding space around the reference position, at each point in time, that the aircraft must stay within to maintain conformance.

As explained in the previous section, the WGS84 geodetic coordinate system will be used as a global standard for specifying reference trajectories. Straight (i.e., minimum distance) segments between geodetic points are great circles in general, but for short segments (away from the earth's poles) a greatcircle is close to linear in latitude and longitude. Geodetic coordinates are inconvenient for specifying and monitoring error tolerances, however. For that purpose, a curvilinear flightpath coordinate system, which follows the assigned trajectory, will be used. An example of a segment of such a curvilinear flightpath coordinate system is illustrated in Figure 1, which shows an explicit along-track/cross-track coordinate grid and the horizontal bounding space.

A curvilinear flightpath coordinate system is a combination of Cartesian and polar coordinate systems.

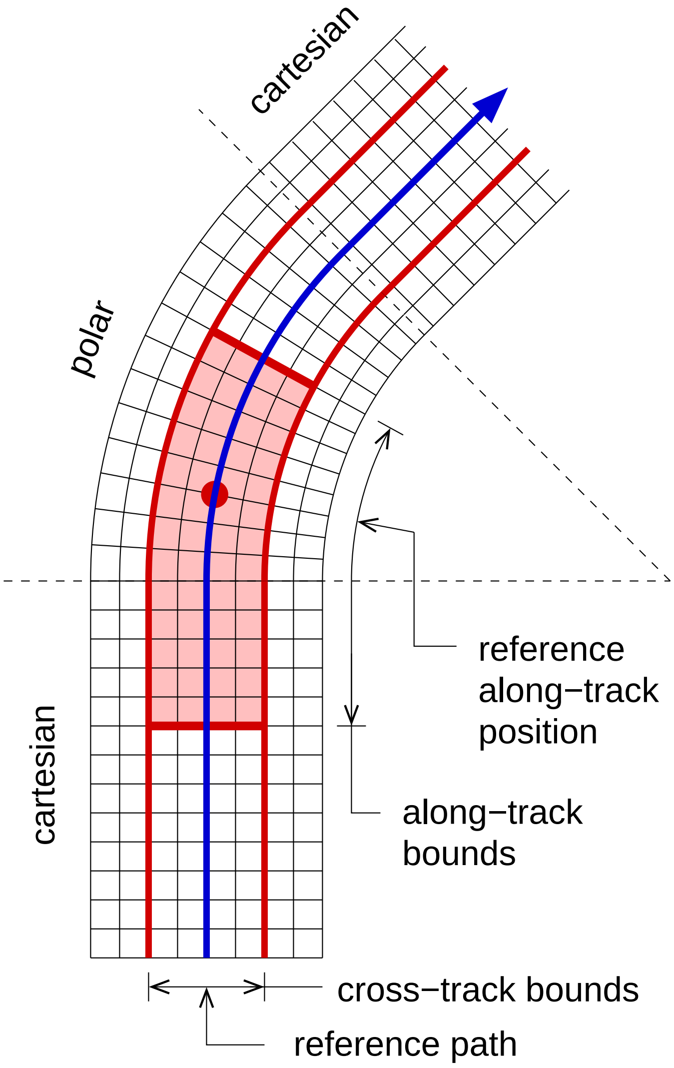

Figure 1. Curvilinear flightpath coordinate system with along-track/cross-track grid and horizontal bounding space (shaded).

The first step in converting from WGS84 coordinates to the curvilinear coordinates is to determine the type of the local coordinate region, which is Cartesian in the (assigned) straight segments and polar (or cylindrical in 3D) in the (assigned) turn segments, as shown in the figure. Actually, these regions are not strictly Cartesian or polar, because they follow the curvature of the earth, but for practical purposes they are Cartesian or polar within the local region of reasonable flight technical errors. The key point is that *each segment defines its own local coordinate system*, which is Cartesian for straight segments and polar for turn segments. Note also that the bounding space is defined in terms of the along-track and cross-track error coordinates. Thus, the bounding space conforms to the curvature of the flightpath, as shown in Figure 1.

Coordinate transformations are needed to transform the geodetic coordinates of an aircraft position to the along-track and cross-track coordinates in the curvilinear flightpath coordinate system. In the straight (greatcircle) segments of the assigned trajectory, the local flightpath coordinate system is approximately Cartesian within the range of practical error tolerances, and the along-track and cross-track coordinates of a point can be determined with established greatcircle algorithms.

The earth is nearly but not quite spherical. The equatorial and polar radii differ by approximately 12 nmi, or about 1/300th of the nominal radius. Greatcircle equations can be based on either a spherical or an ellipsoidal model of the earth. The spherical model yields closed-form analytic solutions, whereas the ellipsoidal model yields more accurate but more complicated iterative algorithms. Cross-track errors are normally a few miles at most and are well approximated with the spherical equations. However, the spherical equations for along-track distances can be off by several miles within the continental U.S., which may be marginally unacceptable for this application, so algorithms based on the ellipsoidal model may be required in practice. For the purposes of this paper, the important greatcircle formulas determine the along-track and cross-track coordinates of a given point, relative to a greatcircle from one given point to another.

The greatcircle equations apply only in the Cartesian-coordinate (straight) regions of the curvilinear flightpath coordinate system. However, they can be adapted for use in the polar-coordinate (turning) regions too. The key is to compute the along-track and cross-track coordinates as if the point were still in the preceding Cartesian region, then convert to polar coordinates. The origin of the polar coordinate system will be the center of the turn arc, and the reference azimuth angle will be at the start of the turn. The actual cross-track coordinate is defined as the radial coordinate minus the nominal radius of the turn, so that the reference cross-track coordinate is always zero (consistent with the straight segments). The along-track coordinate will be the angle from the start of the turn, multiplied by the nominal radius of the turn. Note that if the aircraft is flying the turn with a cross-track error, the actual radius of the turn will be different than the nominal radius, hence the actual along-track distance traveled by the aircraft will be different than the along-track coordinate.

A 4D trajectory also includes a vertical profile describing altitude as a function of time or along-track position. Using along-track position as the independent variable is preferable because it fixes the reference trajectory in the earth-fixed coordinate system. This simplifies conflict calculations and is consistent with standard instrument departures (SIDs), standard arrival routes (STARs), and instrument approach plates, each of which specify any altitude restrictions as a function of position. An assigned trajectory can be visualized as a 3D tube through which the aircraft flies, with the along-track position in the tube constituting the fourth dimension. Specifying altitude as a function of actual along-track position fixes the tube with respect to the earth, whereas specifying it as a function of time would allow it to drift.

Thus, the reference altitude is specified as a function of the *actual* (as opposed to reference) along-track position, as illustrated in Figure 2. The figure shows the reference trajectory as the solid curve with a dot on the curve to indicate the reference position at a point in time. The dashed red lines represent the altitude bounds, which delineate the vertical aspect of the 3D tube mentioned earlier. The other dot in the upper right portion of the figure indicates the *actual* position of the aircraft. The along-track position error is the difference between the actual and reference along-track positions, as shown. Similarly, the altitude error is the difference between the actual and reference altitudes, except that the reference altitude is defined as a function of the *actual*, rather than reference, along-track position, as shown in the figure.

In case this distinction is still unclear, consider the trajectory segment illustrated in Figure 3. If the reference altitude were simply a function of time, then the altitude bounds would be the "wrong altitude bounds" shown in the figure. Rather than being fixed relative to the earth, the 3D tube would effectively shift in space as a function of the along-track error. That would mean that separation might not be guaranteed even if the 3D tubes for two different aircraft were sufficiently separated. The dashed rectangle in the lower right portion of Figure 3 represents the tube for a second aircraft that is flying level into the paper. Based on the properly defined fixed tube, separation is guaranteed regardless of along-track position. Clearly, the "wrong altitude bounds" do not guarantee such separation in this case.

IV. Polynomial Representation of Trajectories

In the current air traffic system, vertical profiles are difficult to predict accurately based on information available to ATC systems on the ground. Part of the problem is that weight and thrust (or throttle setting) are not accurately known by the ground systems. Another major source of altitude prediction error is the uncertainty in the actual time of initiation of altitude transitions. When cleared to climb or descend, the time taken by a pilot to initiate the maneuver can vary by up to nearly a minute. As a result, controllers must reserve a large block of airspace around any aircraft that is in, or is about to enter, an altitude transition. With better information available to ground systems, and with automated piloting, altitude can be assigned more precisely, which will increase airspace capacity.

The objective of specifying a vertical profile is to provide reasonable bounds on altitude without significantly compromising efficiency. The assigned vertical profile should approximate the vertical profile that the aircraft would be most likely to fly normally. Polynomial approximation or curve fitting is a well established

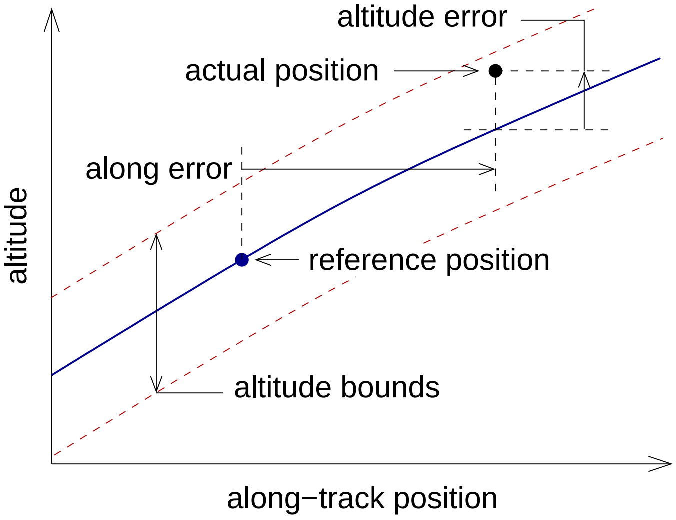

Figure 2. Example showing that reference altitude is a function of *actual* (not reference) along-track position.

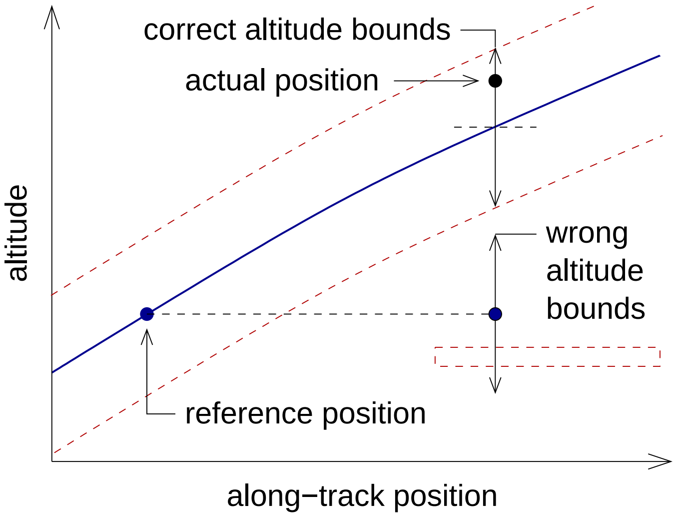

Figure 3. Example showing wrong altitude bounds due to wrong definition of reference altitude as function of time.

and widely used method of data compression that can be applied to this problem. Polynomials have some convenient advantages over discrete points. They are continuous functions, which eliminates the need for interpolation, for example. They can also be differentiated and integrated analytically, which precludes the need for potentially inaccurate numerical algorithms.

In climb and descent, commercial transport airplanes normally fly with the throttle fixed and with feedback to the elevator to maintain constant CAS (at lower altitudes) or constant Mach (at higher altitudes). In the future, the intended CAS/Mach schedule can be downlinked to the ground systems, as can the throttle setting and the estimated weight of the aircraft. The predicted wind, temperature, and pressure fields will be available from a centralized weather data service. Given this data, the vertical profile can usually be predicted fairly accurately, and an approximation of the predicted profile can be used as the reference profile. If the wind data is reasonably accurate, and if the altitude tolerances are reasonable, the reference trajectory can be flown efficiently. Note that altitude tolerances can be a function of traffic density, with looser tolerances

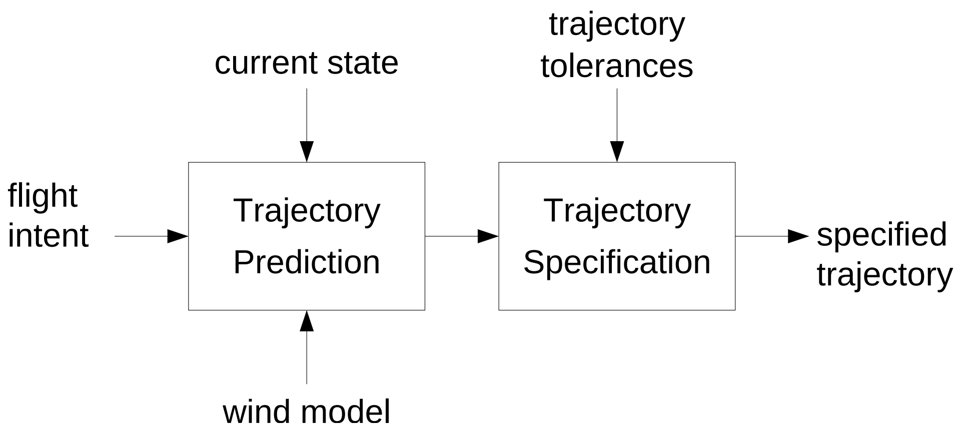

Figure 4. Trajectory specification assigns tolerances to the predicted or reference trajectory.

when density is lower.

Trajectory generation can be similar to what is currently done in the Center/TRACON Automation System (CTAS)[22] to predict trajectories, but will require a few key differences. CTAS is a suite of ATC/ATM decision support tools that is being developed at NASA Ames Research Center. CTAS currently has to guess at the weight and the CAS/Mach schedule to be flown, but those data can be provided by the aircraft or the airline. A more fundamental difference is that the predicted trajectory can actually become the assigned trajectory if it is free of conflicts; otherwise it can be modified to eliminate any conflicts, then become the assigned trajectory. The current ATC system has no such precisely defined vertical profiles. In fact, the notion of vertical conformance itself currently isn't even defined for altitude transition.

The process of generating a trajectory assignment is illustrated in Fig. 4. The trajectory prediction process takes as input the flight intent, the current state, and a wind model. The flight intent comprises the intended horizontal path, speed profile, and target altitude. The predicted trajectory then becomes the reference trajectory and is fed to the trajectory specification process, which assigns tolerances to bound the position of the aircraft at each point in time. The resulting bounded trajectory is then checked for conflicts and becomes the assigned trajectory if no conflicts or other problems are detected. Otherwise the flight intent is modified and the process is repeated until an acceptable trajectory is found.

The CTAS software process that predicts trajectories is called the Trajectory Synthesizer (TS).[23] The TS contains performance models of all major aircraft types, and types that are not modeled directly are approximated with similar available models. The inputs to the TS for each aircraft include the aircraft type and weight, CAS/Mach values, throttle settings, the flightplan, and atmospheric data (winds, temperature, pressure, etc.). The output is the predicted 4D trajectory in the form of a discrete series of points in which the time increment varies with the dynamic state. The TS or its functional equivalent could be used to construct a reference trajectory that is appropriate for each aircraft model. The common trajectory modeling capability currently being discussed by the FAA and Eurocontrol[24] could also eventually be applied to this problem.

Figure 5 shows the altitude profile synthesized by the TS for a constant-CAS climb segment of a Boeing 757 from altitudes of approximately 12,000 to 34,000 ft in a typical wind field. The solid line represents a best-fit parabola, and the dashed lines represent an example error tolerance of ±2000 ft relative to the reference parabola. The parabola should be constrained at the endpoints to match the endpoints of the proceeding and following segments, but that was done here. The constant-CAS segment is followed by a short constant-Mach segment (not shown), which would require its own representation. In most cases, the aircraft should be able to follow the reference trajectory within tolerance by flying the specified CAS of 296 kn as usual. Only if the TS is substantially in error would the aircraft need to use feedback of altitude, and perhaps throttle modulation, to stay within tolerance. Such error could be due to errors in wind, thrust, and/or weight.

The curve fit error bounds of the parabola in Figure 5 are −189 to +289 ft, for a total range of 478 ft. With a vertical error tolerance of ±2000 ft, that fit allows a worst-case altitude deviation, relative to the TS

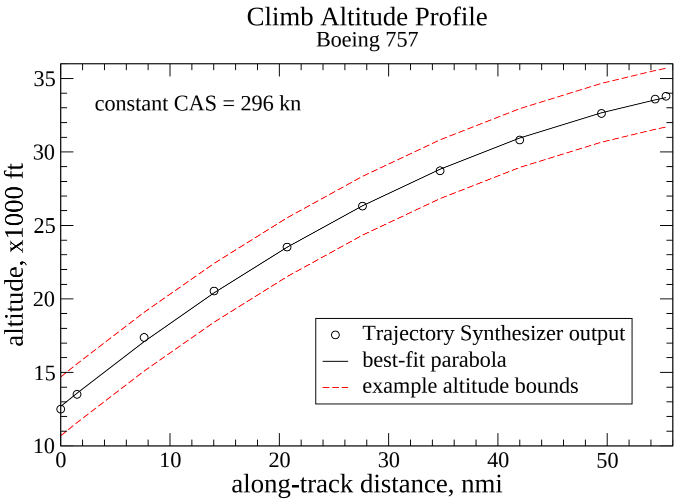

Figure 5. Synthesized altitude profile and best-fit parabola for the constant-CAS segment of a Boeing 757 in climb.

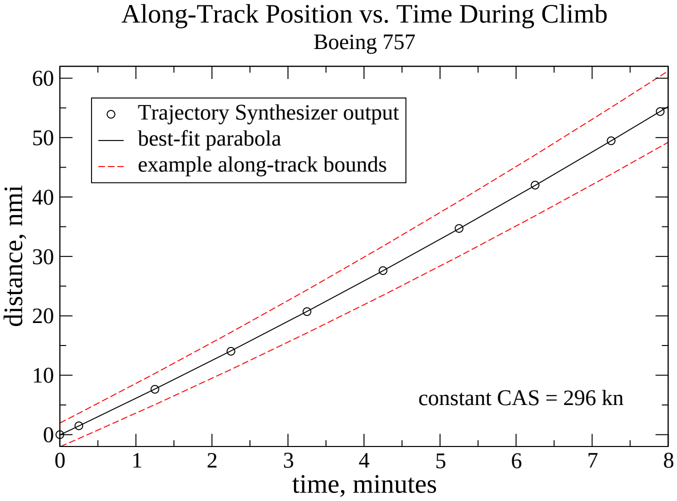

Figure 6. Synthesized along-track position and best-fit parabola for the constant-CAS segment of a Boeing 757 in climb.

output, of 189-2000= -1811 to $2000 - 289 = +1711$ ft, which is probably sufficient. However, if the error tolerance were tighter, say ±1000 ft, then a quadratic fit would only leave a worst-case altitude deviation of -811 to $+711$ ft, which might not be considered sufficient. In that case, the segment could be divided into two or more segments, or a cubic or quartic polynomial could be used for a better fit. For this example, a cubic polynomial fit gives error bounds of -178 to $+94$ ft (272 ft range), and a quartic gives -102 to $+68$ ft (170 ft range). Polynomials of fifth order or higher could have numerical problems and should perhaps be avoided, but polynomials of fourth order or less will not suffer from significant numerical roundoff errors if a consistently high numerical precision of 64 bits is used in both creating and flying the trajectories. The actual order of the polynomials to be used is beyond the scope of this paper, but it could simply start at quadratic and be increased for each case until the required accuracy is achieved. (A constant is sufficient for level flight, of course.)

Figure 6 shows the along-track position associated with the climb of Figure 5. Again, a parabola was generated to fit the TS output, this time the along-track position as a function of time. The resulting error bounds were -0.05 to $+0.07$ nmi, which is close enough for all practical purposes. The example error

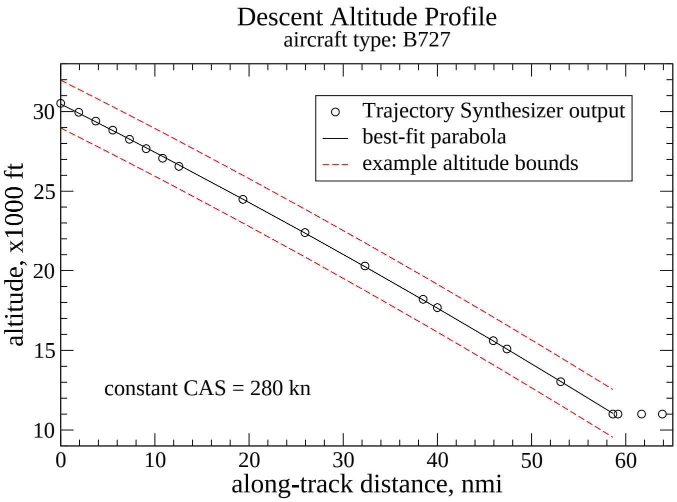

Figure 7. Synthesized altitude profile and best-fit parabola for the constant-CAS segment of a Boeing 727 in descent.

tolerances represented by the dashed lines start out at ±2 nmi and grow linearly with time at a rate of 0.5 nmi/min to ±6 nmi at 8 min from the start of the climb. As an additional safety precaution, the lower bound could be expanded to account for the possibility of reduced thrust due to engine problems.

Figure 7 shows the altitude profile synthesized by the TS for a constant-CAS, idle-thrust descent segment of a Boeing 727 from altitudes of approximately 30,000 to 11,000 ft in a typical wind field. Again, the solid line represents the best-fit parabola, and the dashed lines represent a hypothetical error tolerance of ±1500 ft. The constant-CAS segment is preceded by a short constant-Mach segment (not shown), which would require its own curve fit. Again, the aircraft should normally be able to fly the constant CAS of 280 kn without altitude feedback or throttle modulation and stay within the specified altitude range. As before, altitude feedback, and perhaps throttle modulation, could be activated when the altitude deviation reaches some threshold value. With error bounds of -53 to $+116$ ft, the curve fit for this descent is much more accurate than for the climb of Figure 5. Descents tend to be more nearly linear than long climbs, and are usually well modeled with a parabola. In general, an arrival descent would be followed by a short level cruise segment into the meter fix, which would allow the aircraft to cross the meter fix at a precise level altitude.

The along-track position associated with the descent of Figure 7 is not shown, but it would be similar to figure 6 for the climb example, except that the error tolerances might decrease rather than increase with time if the aircraft is required to arrive at a meter fix at a precise time.

V. Proposed XML Format

The purpose of XML (Extensible Markup Language) is to create standards for data specification and transfer. Unlike its more specialized sibling HTML (Hyper-Text Markup Language), XML allows standards designers to define their own data structures. XML is rapidly replacing binary formats for automated business-to-business transactions and is being widely used for computing standards such as Scalable Vector Graphics (SVG). XML provides flexibility in the selection and ordering of the data fields, which is indispensable for trajectory specification because each trajectory can have a variable number of segments of various types. The flexibility also allows trajectories to be updated without repeating all of the data that remains unchanged from the previous update.

XML is not required for this application, but it's versatility, standardization, and growing popularity seem to make it a good choice. Note that real-time decision makers such as pilots and controllers will not read or write XML text directly but can be provided with simplified graphical representations of trajectories and a "point-and-click" interface where needed. Note also that message integrity can be guaranteed by using a secure hash algorithm (e.g., SHA-1) and a "handshaking" verification procedure.

The structure and form of an XML document can be formally described by another XML document called a Schema. Alternatives to Schema are also available. The objective of this section, however, is not to formally define an XML format but rather to suggest how the format might look and what information it should contain. Example XML code will be presented and discussed in sufficient detail to provide high-level design requirements for a formal specification.

At the most basic level, an XML document consists of a hierarchy of elements, each of which can contain subelements and/or attributes. Consider, for example, the following XML fragment:

```
<elem attr="yes">
    <sub attr2="100" attr3="no"/>
    </elem>
```

The main delimiters in XML are the angle brackets, “`<`” and “`>`,” which enclose the opening and closing tags of each element or subelement. The example shows an element called “`elem`,” which has an attribute called “`attr`” and a subelement called “`sub`.” All attributes are specified in the opening tag of an element, and the closing tag contains the element name preceded by a forward slash, such as “`</elem>`” above. Elements that contain only attributes and no subelements can end the opening tag with “`/>`” in place of a separate closing tag, as shown in the example. Attribute values must always be in quotes, and the allowed values can be restricted to a specified discrete list. Attribute values can also be restricted to specified types, such as character string, integer, and decimal number.

In this application, XML will be used to specify several physical quantities such as time, distance, speed, weight, etc. The units could be specified explicitly, but for simplicity a set of standard aviation units will be used as the default in most cases. The default unit for horizontal distance will be nautical miles (nmi), for example, and for altitudes the default unit will be 100 ft. Time will be specified in the standard XML format of “hh:mm:ss” (two digits each for hours, minutes, and seconds). The complete set of default units is shown in Table 1. Alternative units could be allowed to override these defaults if desired, but that will not be discussed in this paper.

Table 1. Default physical units

quantity	unit
time	hh:mm:ss (XML time format)
horizontal distance	nautical miles (nmi)
altitude	100 feet (100 ft)
angles	degrees (deg)
horizontal speed	knots (kn)
vertical speed	feet/minute (ft/min)
weight	1000 pounds (klbs)

```
<flight ...>
    <aircraft ...> ... </aircraft>
    <preferences> ... </preferences>
    <trajectory ...> ... </trajectory>
    </flight>
```

XML Sample 1. Top-level structure

At the top level, the proposed XML trajectory specification format appears as shown in XML Sample 1. Ellipses (“`...`”) represent text that has been omitted for simplicity. The root element is “`flight`,” and it contains the top-level elements “`aircraft`,” “`preferences`,” and “`trajectory`.” Note that these element names (and those to follow) could be abbreviated if datalink bandwidth is a problem, but full names will usually be used in this paper for clarity. The “`aircraft`” element gives information about the aircraft itself. The “`preferences`” element provides information about the airline’s or pilot’s preferred flight parameters. Finally, the “`trajectory`” element specifies the trajectory itself. Each of these elements will be discussed in more detail below. Note that the “`aircraft`” and “`preferences`” elements can be specified once and need not be repeated each time the trajectory is revised, unless they are revised too.

```
<flight ID="AAL2332/SFO" dest="JFK"
    date="2004-02-04" time="13:25:00"
    bcode="2187" rev="0.0.0"
    status="request">
```

XML Sample 2. Flight element attributes

The root element “`flight`” has several attributes, as shown in XML Sample 2. The “`ID`” attribute gives the standard flight identification number with the originating airport code (for the current leg of the flight) appended after a slash. The originating airport could be another attribute, but appending it to the flight identification number helps prevent confusion with previous or subsequent legs of the same flight (which could be in the system at the same time). The “`dest`” attribute gives the destination airport code. The “`date`” and “`time`” attributes specify the scheduled departure date and time. The “`bcode`” attribute gives the aircraft transponder beacon code. The “`rev`” element gives the revision number of the trajectory in a format to be discussed later.

The “`status`” attribute tells whether the information to follow is a “`request`” or is actually “`assigned`.” Other status types might also be useful, depending on the concept of operations. For example, a status of “`mandatory`”might apply when an imminent conflict is being resolved. The question of whether or when a pilot has veto power over an assigned trajectory is an operational issue that is outside the scope of this paper. Note, however, that if a trajectory is tentatively assigned but pending approval by the pilot, then both the tentative trajectory and the active trajectory need to be kept clear of new conflicts (as a result of new trajectory assignments to other aircraft) until the pilot decides whether or not to accept the assignment.

The pilot or airline need not specify an actual trajectory if they are not equipped, or do not wish, to do so. They can simply specify their origin, destination, aircraft type, and, optionally, their flight preferences, then let the ground system specify and assign a trajectory. In that case, the pilot or airline would use the “`aircraft`” and “`preferences`” elements and omit the “`trajectory`” element. A revision number of “`0.0.0`” could apply in that case.

When a trajectory is initially assigned, the revision number in the “`rev`” attribute can be set to “`1.0.0`,” and the “`status`” attribute can change from “`request`” to “`assigned`.” The initial assigned trajectory can be fully specified from start to finish, or it can be fully specified for, say, the first hour, and only the horizontal route tentatively specified for the remainder of the flight, pending later, more precise specification. When a trajectory is actually assigned, the root element “`flight`” will have another attribute called “`assigntime`,” which gives the assignment uplink time.

```
<flight ID="AAL2332/SFO" CID="324459"
    assigntime="14:05:32"
    devtime="14:09:52" rev="1.0.2"
    status="assigned">
```

XML Sample 3. Flight element attributes for trajectory revision

When a trajectory is revised, a trajectory deviation time will also be specified in an attribute called “`devtime`.” The revised trajectory must be continuous with the old trajectory so the aircraft can maintain continuous conformance, and “`devtime`” specifies the time at which the new trajectory actually deviates from the old. The deviation time must follow the assignment time by a sufficient margin to allow the new trajectory to be uplinked, accepted, and processed onboard the aircraft. The determination of that margin is an operational consideration outside the scope of this paper. An example of the “`flight`” tag for a trajectory revision appears in XML Sample 3. Note that the scheduled departure date and time need not be repeated because they are constant, and the same applies to the destination airport and beacon code, assuming they haven’t changed. However, the flight identification and the computer identification number should be given for positive identification.

A. Aircraft

```
<aircraft tail="N788" model="MD80">
    <weight unit="klbs" value="135"/>
    <fuel unit="gal" amount="5226"/>
    <engine model="JT9D" factor="0.98"/>
    <equip code="GAF" status="normal"/>
    </aircraft>
```

XML Sample 4. Aircraft element

As mentioned above, the “`aircraft`” element is a top-level subelement of the root element that can be used by the airline or pilot to downlink the aircraft model and parameters. (This element could conceivably be sent in advance via landline to reduce wireless bandwidth usage.) An example is shown in XML Sample 4. The “`tail`” attribute of the “`aircraft`” element identifies the tail number of the aircraft. The “`model`” attribute identifies the aircraft performance model, which will be selected from an approved list. The ground systems will have performance models of each aircraft type, which can be used to construct an efficient trajectory when a fully specified trajectory request is not received from the aircraft or airline.

The “`weight`” element specifies the takeoff weight of the aircraft in units specified by the “`unit`” attribute. The units of “`klbs`” (1000 lbs) shown in the example could be the default, and other options such as “`kg`,” for kilograms, might be allowed. The “`fuel`” element gives the amount of fuel stored at takeoff. It could also be used to update the fuel level at a particular time if desired, in which case an additional time attribute could be used. The “`engine`” element has attributes “`model`” and “`factor`,” which specify the engine model and an optional thrust factor, which defaults to “`1.0`.” The thrust factor scales the nominal maximum thrust for that engine model, and it could be used to provide a more precise maximum thrust for that particular engine, if known. The “`equip`” element gives an avionics equipment code and a functional status code that could be considered optional if everything is functioning properly. Status codes would need to be agreed upon, but they are beyond the scope of this paper.

B. Flight Preferences

As mentioned above, the “`preferences`” element is a top-level subelement of the root element that can be used by the airline or pilot to downlink the preferred flight parameters. (This element could also be sent in advance via landline.) It can provide ground systems with the basic parameters necessary to construct an efficient trajectory consistent with the airline or pilot preferences, if necessary. An example is given in XML Sample 5.

```
<preferences>
    <climb thrust="90" Mach="0.74"
        CAS="280"/>
    <descent thrust="5" Mach="0.74"
        CAS="290"/>
    <cruise alt="310" Mach="0.76"/>
    <turn radius="9.0"/>
    <depart gate="12B" runway="10L"/>
    <arrive gate="8A" runway="22L"
        order="14"/>
    <route>xxx.xxx,xxx.xxx
        xxx.xxx,xxx.xxx
        xxx.xxx,xxx.xxx
        xxx.xxx,xxx.xxx</route>
    </preferences>
```

XML Sample 5. Preferences element

The “`climb`” and “`descent`” elements each have a “`thrust`” attribute to specify a preferred thrust power setting in percent of maximum thrust. Climbs and descents usually consist of a long constant-CAS (Calibrated Airspeed) segment at low altitudes and, for jets, a constant-Mach segment above the CAS/Mach transition altitude. The “`climb`” and “`descent`” elements each have “`CAS`” and “`Mach`” attributes to specify the constant CAS and constant Mach to be flown. For additional flexibility, a time-varying CAS or Mach could be allowed during altitude transition, and it could be specified as a polynomial function of time, but that option will not be discussed here. The “`cruise`” element specifies the preferred cruising altitude and Mach or CAS.

The “`turn`” element has a “`radius`” attribute to specify the preferred default turn radius. Alternatively, a maximum bank angle for a coordinated turn could be specified using a “`bank`” attribute. The “`depart`” and “`arrive`” elements each have a “`runway`” attribute to make known the preferred runways. They also each have an optional “`order`” element to specify the preferred takeoff or landing order relative to other flights of the same airline company. The optional “`gate`” element could be of use for surface traffic optimization. Finally, the “`route`” element can be used to specify the desired horizontal route waypoints in terms of WGS84 (latitude and longitude) points. Each waypoint could be a comma-separated pair, and the waypoints could be separated by spaces, the format used to represent polygon vertexes in the XML standard for Scalable Vector Graphics (SVG).

C. Trajectory

The actual trajectory is specified in the “`trajectory`” element, a top-level subelement of the root element. It could be specified by the aircraft as a request, or it could be specified by ground systems as an assignment. The “`trajectory`” element consists of several subelements including “`segments`,” which specifies the actual trajectory in terms of individual “`segment`” elements, each of which specify a trajectory segment. An example is shown in XML Sample 6.

The “`trajectory`” element has attributes “`reftime`” and “`wthr`.” The “`reftime`” attribute specifies the trajectory reference time, relative to which all other times in the trajectory will be specified. By changing this reference time, the entire trajectory can be shifted in time when necessary, which will be discussed later. The zero reference for along-track position will correspond to the reference position of the aircraft at the trajectory reference time. As the flight progresses, past segments can be dropped and the reference time can be moved up to the beginning of the current trajectory segment, if desired. Also, the hours field can

```
<trajectory reftime="13:46:17"
    wthr="2004-02-04T13:00">

    <constraint> ... </constraint>
    <arrive runway="10L" gate="12B"/>

    <tolerances>
        <cross tol="2.0"/>
        <vert tol="2"/>
        <along tol="-2.0 2.0"
            rate="-10 10" time0="0:32"
            max="-10 10"/>
        </tolerances>

    <segments>
        <segment ...> ... </segment>
        <segment ...> ... </segment>
        ...
        <segment ...> ... </segment>
        </segments>

    </trajectory>
```

XML Sample 6. Trajectory element

be allowed to go past 24 if the flight continues past midnight of the day it started. The "`wthr`" attribute specifies the exact weather data that was used in the construction of the trajectory.

The optional "`constraint`" element can be used by ground systems to specify an arrival fix (3D position) and time constraint when arrival metering is in effect. Arriving flights need to know this constraint if they intend to construct their own descent trajectory when arrival metering is in effect. The "`constraint`" element could also be used for traffic flow management by specifying a long-range airport arrival time or by specifying the polygon vertices of flow constrained areas (FCA) for a specified period of time. The details of such constraints are beyond the scope of this paper. If more than one constraint is necessary, a separate "`constraint`" element can be used for each one, and a constraint number should be specified for each to prevent any confusion or ambiguity about whether an additional constraint is being imposed or an existing constraint is being revised.

The "`arrive`" element has an attribute called "`runway`" to specify the arrival runway and an optional "`gate`" attribute to specify the arrival gate, which could be useful for surface traffic management. Similarly, a "`depart`" element (not shown) could be used before takeoff to specify the departure runway and gate.

The "`tolerances`" subelement of "`trajectory`" will be used by ground systems to uplink the default trajectory tolerances. It contains "`along`," "`cross`," and "`vert`" subelements to specify the default along-track, cross-track, and vertical tolerances, which can be overridden in each trajectory segment. Each of those elements has a "`tol`" attribute that is used to specify the actual tolerances. The default cross-track tolerance will always be symmetric left and right, so only a single value is given, in the default units of nautical miles. The default vertical tolerance will apply only to level flight, and it will also be symmetric up and down, so a single value is given in the default units of 100 feet. The tolerances will be limited by RNP performance and may need to be looser for aircraft with with a lesser RNP capability. Better-equipped aircraft can perhaps be given certain preferences (in conflict resolution, for example) as an incentive to equip to higher levels, but such preferences are outside the scope of this paper.

The requirements for along-track tolerances are slightly more complicated. Unexpected along-track winds (headwinds or tailwinds) can cause significant differences between predicted groundspeed and what can be flown efficiently. Predicted groundspeed is the sum of the airspeed setting and the predicted wind speed. If the predicted wind speed is significantly in error, the airspeed will need to be adjusted to achieve the predicted or assigned groundspeed. But fuel efficiency is a function of airspeed, and the range of flyable airspeeds is obviously limited. Tighter along-track tolerances can increase airspace capacity, but they also increase the probability that inefficient or unflyable airspeeds will be demanded. Along-track position error tolerances need to be set as a compromise between those two effects.

A constant difference between actual and predicted groundspeed causes the along-track position error to increase linearly with time. To allow for an effective tolerance on groundspeed, along-track position tolerances should be allowed to vary linearly with time. Also, because optimal airspeed is usually closer to maximum than to minimum airspeed, speedup and slowdown capabilities relative to optimal airspeed are usually asymmetric. Hence the along-track tolerances should also be allowed to be asymmetric.

The "`tol`" attribute of the "`along`" element, therefore, has two values: the forward and rear tolerances, in units of nautical miles. These are the initial tolerances at the time specified in the "`time0`" element, which is relative to the trajectory reference time specified in the "`reftime`" attribute of the "`trajectory`" element discussed earlier. The actual tolerances vary as a function of time at a rate specified in the optional "`rate`" attribute, if specified, which also has two values, in units of knots, that default to zero. However, the maximum magnitude of the along-track tolerances is capped by the optional "`max`" attribute, if specified. Note that the along-track assigned position, tolerances, and speed can be updated periodically, which will be discussed later in the paper.

D. Trajectory Segments

The actual trajectory is specified in the "`segments`" subelement of the "`trajectory`" element. It contains a series of "`segment`" subelements, each of which specifies a segment of the trajectory. An example of a "`segment`" subelement is given in XML Sample 7.

```xml
<segment number="1" vtype="climb"
    htype="straight" stype="constCAS">

    <time start="0:08:42" duration="7:42"/>
    <begin coords="WGS84" lat="xxx.xxxx"
        lon="xxx.xxxx"/>
    <end coords="WGS84" lat="xxx.xxxx"
        lon="xxx.xxxx"/>
    <along coeffs="xxx.xxx xxx.xxx"
        CAS="280" length="27.815"/>
    <alt coeffs="126.8 21.609 4.1417e-3"
        thrust="90" end="270" max="272"/>
    <tolerances>
        <along rate="-15 15"/>
        <vert tol="-15 10"
            rate="-1.5 1"/>
        </tolerances>
    </segment>
```

XML Sample 7. Trajectory segment element

The "`number`" attribute of "`segment`" gives the segment number in the sequence. Although the segments should obviously be listed in chronological order, the segment number provides a handy reference tag and minimizes the chance of confusion. The "`vtype`," "`htype`," and "`stype`" attributes specify the type of the segment, which determines which other subelements apply. The "`vtype`" element specifies the vertical type, "`htype`" specifies the horizontal or heading type, and "`stype`" specifies the speed type. The allowed types are listed in Table 2. The vertical types are climb, level, and descent. The heading types are right turn, left turn, and straight. The speed types are constant CAS, constant Mach, speedup (increasing CAS or Mach), slowdown (decreasing CAS or Mach), and dependent. Note that the speed types are based on airspeed (CAS or Mach) rather than groundspeed, which changes during altitude transition even when CAS or Mach are constant with no winds. Whether speedup or slowdown refer to CAS or Mach will depend on which is specified, as will be discussed shortly. The dependent type is a catch-all for cases where CAS or Mach could vary depending on other requirements, such as a constant flightpath angle for climb or descent. The total number of permutations of possible segment types is $3 \times 3 \times 5 = 45$, though some will rarely or never be used. The rationale behind these trajectory segment types will be discussed in more detail later.

Table 2. Trajectory segment types

name	description
`vtype`	vertical type
`climb`	increasing altitude
`level`	constant altitude
`descent`	decreasing altitude
`htype`	heading type
`rturn`	right turn
`lturn`	left turn
`straight`	greatcircle path
`stype`	speed type
`constCAS`	constant CAS
`constMach`	constant Mach
`speedup`	increasing CAS or Mach
`slowdown`	decreasing CAS or Mach
`dependent`	dependent CAS or Mach

The “`time`” subelement of “`segment`” specifies the precise time range of the segments. The “`start`” attribute gives the starting time of the segment relative to the trajectory reference time, and the “`duration`” attribute gives the time duration of the segment. The “`begin`” and “`end`” subelements of “`segment`” specify the exact ground position of the beginning and end of the segment. The attribute “`coords="WGS84"`” shown in the example indicate that the coordinate are latitude and longitude in the WGS84 geodetic reference system, which could be considered the default. (The beginning of a segment must match the end of the previous segment, so it is redundant information that could perhaps not be required if bandwidth is an issue.)

Local coordinate systems could also be convenient in terminal areas around major airports. A standard stereographic (pseudo-Cartesian) coordinate system can be defined for each major airport. The “`coords`” attribute can be the airport code, and the coordinates will be specified, in units of nautical miles, with “`x`” and “`y`” attributes, as they are traditionally called. For example:

```
<begin coords="DFW" x="34.344" y="9.439"/>
```

All straight segments will be assumed to follow greatcircles, although for short segments the difference between a greatcircle and a straight line in stereographic coordinates is very small. Finally, conventional named waypoints should probably still be allowed, and will be designated with “`coords="named"`.” For example:

```
<end coords="named" name="KARLA"/>
```

Named waypoints could be useful for arrival meter fixes, for example. However, care must be taken to ensure that everyone always agrees on the actual positions (latitude and longitude) of named waypoints. That could be a challenge in itself if the positions change occasionally, as they do currently.

Referring back to XML Sample 7, the “`along`” subelement of “`segment`” specifies the along-track position as a polynomial function of time. The “`CAS`” attribute gives the nominal CAS that is expected to be flown.

It could be replaced by a “`Mach`” attribute when appropriate, of course. The “`length`” attribute gives the along-track length of the segment. The “`coeffs`” attribute lists the coefficients of the polynomial in order of increasing powers, starting with the constant term. The distance units will be nautical miles for all coefficients. The linear coefficient should be in standard units of knots (nmi/hr) for readability, but the succeeding coefficients should probably be in units of kn/min, kn/min/min, etc., for better scaling. Consistent units are desirable, but readability and reasonable scaling are more important. As long as the units are clearly specified in the standard, the mixing of hours and minutes should not be a problem. The value of time used as the argument of the polynomial will be the time relative to the start of the segment. Thus, the first (constant) coefficient will give the along-track (scalar) starting position of the segment, where the zero reference for along-track position corresponds to the assigned reference position of the aircraft at the trajectory reference time. The second coefficient is the groundspeed, in knots, at the start of the segment.

Referring back again to XML Sample 7, the “`alt`” subelement of “`segment`” specifies the altitude as a function of along-track position (a scalar). The “`thrust`” attribute gives the nominal thrust for the climb segment, in percent of full power, and the “`end`” attribute gives the assigned altitude at the end of the segment. The “`max`” attribute gives an upper altitude limit that overrides the vertical tolerance near the end of the climb, as will be discussed later. The “`coeffs`” attribute gives the coefficients of the altitude as a function of *actual* (not reference) along-track position, as was illustrated in Fig. 2. As with the “`along`” element, the coefficients will be listed in order of increasing powers. The distance used as the argument of the polynomial will be the along-track position, in nautical miles, relative to the start of the segment. Thus, the first (constant) coefficient will be the altitude, in hundreds of feet, at the start of the segment. The second coefficient will be the altitude rate at the start of the segment, in units of 100 feet/minute.

An interesting side issue is whether altitude should be specified in terms of geometric altitude or pressure altitude. In the United States, pressure altitude is currently used above the transition altitude of 18,000 ft, and aircraft performance charts are based on pressure altitude. However, geometric altitude can be determined by GPS/WAAS (Global Positioning System with Wide Area Augmentation System) much more accurately than pressure altitude can be measured. GPS/WAAS can also determine geometric altitude *rate* much more accurately than pressure altitude rate can be determined. Accurate altitude and altitude rate determination are indispensable for rapid detection of trajectory nonconformance or impending nonconformance. Accurate altitude control is also very desirable for the “continuous altitude rule” that was proposed in reference.[25] This rule designates cruising altitudes as a continuous function of course angle, which spreads cruising traffic vertically, greatly reducing the chance of collision. For these reasons, geometric altitude should eventually replace pressure altitude as the standard for altitude assignment.

Referring back yet again to XML Sample 7, the “`tolerances`” subelement of “`segment`” will be used by ground systems to uplink trajectory error tolerances for the segment. It has the same basic format as the “`tolerances`” subelement of the “`trajectory`” element one level up in the XML tree, which was used to specify the *default* tolerances for all segments. If no tolerances are specified for a particular segment, the defaults apply. If tolerances are specified for a particular segment, they apply only to that segment. For climb and descent segments, default altitude tolerances are not provided, so they need to be specified for each trajectory.

```
<tolerances>
    ...
    <along type="relative" ID="DAL257"
        tol="-1:50 -1:30"/>
    </tolerances>
```

XML Sample 8. Relative along-track tolerance

During climb and cruise, the along-track position error tolerance will typically be allowed to expand with time in both directions, which means that the lower bound of the along-track tolerance rate will be negative and the upper bound will be positive. That could be reversed for descent, however, if a precise arrival time is required at a meter fix. For intrail arrivals on the same flightpath, along-track tolerances could conceivably be relative to the preceding aircraft. This would obviously require that the following aircraft be equipped to track the leading aircraft. Such a relative along-track tolerance could be specified, for example, as shown in XML Sample 8. The “`ID`” attribute specifies the flight to be followed, and “`tol="-130 -1:50`”:” would be interpreted as requiring a following time between 1:30 and 1:50. In other words, the following aircraft would be required to pass any given along-track point at some time between 1:30 and 1:50 later than the leading aircraft.

Also, for non-level (climbing or descending) segments, the vertical tolerances can be allowed to vary linearly with time and to be asymmetric for enhanced flexibility. Hence, the “`vert`” subelement has a “`tol`” attribute that specifies both a lower and an upper tolerance in units of 100 feet, and it also has a “`rate`” attribute that specifies the rate of change of those tolerances in units of 100 feet per nautical mile. The along-track reference position for vertical segments is the beginning of the segment. This is different from the along-track reference time, which can specified independently of any particular segment, as discussed earlier. Note that linear variation with time could also be allowed for the cross-track position tolerance if deemed appropriate.

The “`along`” subelement of “`segment`,” as shown in XML Sample 7, applies to straight (greatcircle) segments. For turning segments, all the same elements apply except “`along`,” which is replaced with “`turn`.” An example of a turn segment is shown in XML Sample 9. The elements in common with all segment types are shown in abbreviated form for simplicity.

```
<segment number="1" vtype="level"
    htype="rturn" stype="constMach">

    <time .../>
    <begin .../>
    <end .../>
    <alt .../>

    <turn begin="84.6" angle="7.3"
        end="91.9" radius="15.0">
        <wind speed="46" dir="243"/>
        </turn>

    <tolerances> ... </tolerances>
    </segment>
```

XML Sample 9. Turn segment

The “`turn`” subelement has attributes “`begin`,” “`end`,” “`angle`,” and “`radius`.” The “`begin`” and “`end`” attributes specify the course angle at the beginning and end of the turn, and the optional “`angle`” gives the angle of the turn, where positive is to the right (“`angle`” would be for the convenience of the human reader). Note that course angle is the angle of the groundtrack, independent of winds, where zero is due North and 90 deg is due East. The “`radius`” attribute gives the radius of the turn in units of nautical miles, as usual, and it should probably be limited to something like 100 nmi. The “`turn`” element has one subelement, “`wind`,” with attributes “`speed`” and “`dir`” to specify the speed and direction of the wind in units of knots and degrees, respectively. Since turns are relatively short in duration, the wind field will be assumed to be uniform throughout the turn. These parameters precisely determine the horizontal position of the aircraft as a function of time throughout the turn.

The wind vector is added to the airspeed vector to determine the varying groundspeed of the aircraft as it progresses through a turn. The law of cosines can be used to compute the resulting groundspeed, which can be numerically integrated to precisely determine the along-track position as a function of time. In any non-

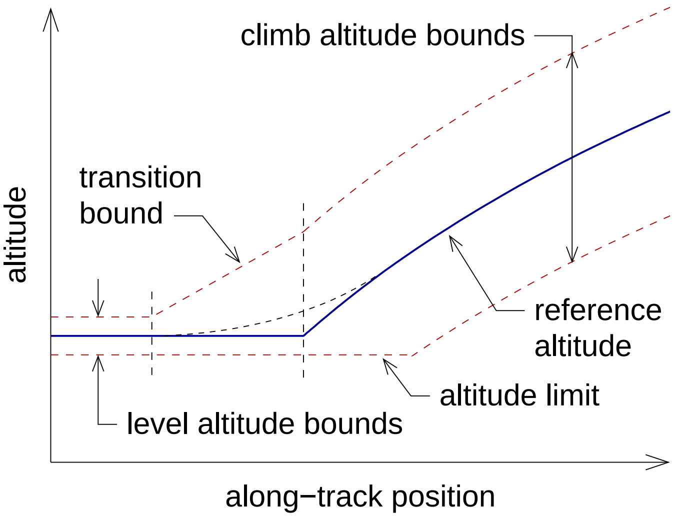

Figure 8. Transition bound to accommodate vertical dynamics at start of climb (or descent) without requiring another explicit segment.

zero wind field the groundspeed will vary through the turn, and the bank angle must be varied accordingly to maintain a coordinated turn (in which gravity and centrifugal acceleration add vectorially to a force normal to the floor of the aircraft). The necessary bank angle, ϕ, is given in terms of the varying groundspeed, v, by $\phi = \operatorname{atan}(v^2/(rg))$, where r is the (constant) turn radius, and g is gravitational acceleration. The radius of the turn can be selected so that the maximum bank angle through the turn does not exceed a specified magnitude.

A few additional points are worth mentioning with regard to trajectory segment types listed in Table 2. Higher-order turn dynamics could be modeled with additional trajectory segment types, but the small improvement in accuracy they would provide is not likely to justify the added complexity. Segment types could be added to model the roll dynamics at the start and end of turn segments, for example, but those roll dynamics last only a few seconds and their effect can simply be absorbed into the cross-track error tolerance. Turns of small magnitude should probably be given a large radius (but not more than, say, 100 nmi) to minimize the required bank angle and the resulting modeling error due to the roll dynamics. Note also that an FMS can compensate for the effective delay simply by starting the turn a few seconds early. The turn lead time is approximately half of the time needed to bank over (bank angle divided by nominal roll rate).

The normal (vertical) acceleration dynamics at the start and end of altitude transitions are another matter. Those dynamics can last approximately ten to twenty seconds, and their effect on the trajectory can be significant. The same sort of lead compensation discussed in the preceding paragraph for roll dynamics can be used, but if that is not accurate enough, a short climbing or descending segment of ten to twenty seconds can be used to model the vertical acceleration dynamics. Figure 8 shows such a segment as the short dashed line rounding the corner at the start of the climb. But rather than modeling this short segment explicitly, a simpler approach is to define a transition bound, as shown in the figure, to allow a larger altitude tolerance at the start of climb (or descent).

To be valid, segments must be labeled with the correct type. For example, level segments must have a constant altitude, climb segments must have a positive altitude rate, and descent segments must have a negative altitude rate. Also, position, groundspeed, and course (groundtrack) angle must all be continuous to form a valid trajectory. This means that a turning segment is required between any two segments for which the course angle does not match at the end of the first segment and the beginning of the second. Similarly, a speedchange segment is required between any two segments for which the groundspeed does not match at the end of the first segment and the beginning of the second. A small discontinuity should be allowed in the vertical speed, however, at the beginning and end of altitude transitions and at CAS/Mach transitions.

The turn angle between adjacent straight (greatcircle) segments can be determined using a standard "course" function, where "course(A,B)" is defined as the initial course angle of the greatcircle from point

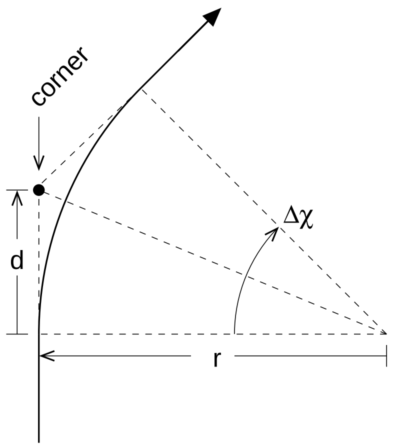

Figure 9. Turn geometry: $d = r\tan(\Delta\chi/2)$.

A to point B (the spherical-earth equations are plenty accurate for this purpose). The turn angle between greatcircles (A,B) and (B,C) is "course(B,C) − course(B,A) ± 180 deg." Each turn must be tangent to the two greatcircle segments that it connects. The turn corner is at the intersection of the two greatcircle segments, and a turn of angle $\Delta\chi$ must start at a distance $d = r\tan(\Delta\chi/2)$ before the corner, as shown in Figure 9.

The 45 combinations of segment types cover all normal flight modes. The types involving turns and/or airspeed changes will typically be of relatively short duration compared to the straight segment types with constant airspeed. The segment types involving a simultaneous turn and speed change will probably be used rarely but are available when needed. These segment types may seem to preclude certain combinations, such as a turn that continues through a transition from climb to level flight, for example. Note, however, that such a combination could be constructed, if necessary, by simply following a climbing turn segment with a level turn segment. Note also that sequential turn segments could be used for S-turn delays and other unusual maneuvers.

The 45 different possible combinations of segment types are not all parametrically distinct. Climb and descent segments have the same parameters, for example, and differ only by the the sign of the altitude rate. Similarly, level segments are just a special case in which the altitude rate is zero. The climb, level, and descent designations are therefore not necessary for the actual construction of the trajectory. They are essentially for the convenience of the human reader. The only distinction as far as parametrization is concerned is the distinction between turning and straight segment types. The turning segment types need a few additional parameters such as the radius of the turn and the wind vector, which were discussed earlier.

A problem with transitioning from a non-level segment to a level segment is that the altitude tolerance will almost always be discontinuous. A typical altitude tolerance for level flight might be ±200 ft, but for climb or descent the tolerance could be ten times or more larger. Going from a level segment to a non-level segment is not a problem because the tolerance increases, but going from a non-level segment to a level segment is a problem because the altitude tolerances decrease sharply and discontinuously, as shown in Figure 10. The linearly decreasing "transition tolerance" shown for the lower altitude tolerance is one possible approach for reducing the tolerance less abruptly. The along-track distance over which the tolerance decreases could default to something like 5 or 10 nmi or could be specified explicitly, if desired, in the "`ttdist`" ("tolerance transition distance") subelement of the "`segment`" element for the level segment. Note that "`ttdist`" could also apply to the cross-track tolerance in transitioning from a turn segment to a straight segment.

Figure 10 also illustrates another problem with transitioning from a non-level segment to a level segment. The upper altitude tolerance during the climb segment allows the aircraft to go significantly above its intended cruising altitude, exposing it to potential conflicts with traffic at the next higher flight level. To prevent such exposure, an "altitude limit" can be specified in the "`max`" attribute of the "`alt`" subelement of "`segment`." This maximum altitude limit overrides the upper altitude tolerance, as shown in the figure. For a descent segment, the "`max`" attribute would be replaced by "`min`." This constraint prohibits any "overshooting" of the target leveloff altitude and will normally be the upper altitude tolerance for the level segment. The same

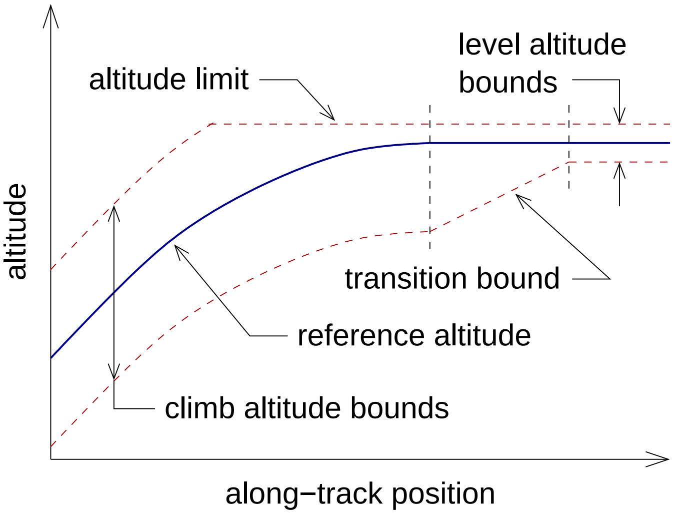

Figure 10. Altitude limit and transition tolerance for typical transition from climb to level segment with decreasing altitude tolerances.

geometry flipped upside down applies to descent.

Also, as mentioned earlier, the "`rev`" attribute of the root element "`flight`" gives the trajectory revision number in the form "`i.j.k`," where "`i`" is the horizontal path revision number, "`j`" is the vertical profile revision number, and "`k`" is the along-track update number. The revision number can start as "`1.0.0`" for the first assigned trajectory. If the horizontal path is revised, then "`i`" will be incremented and "`j`" and "`k`" will each be reset to zero. Similarly, if the vertical profile is updated (as in a temporary altitude assignment to resolve a conflict), then "`j`" will be incremented, and "`k`" will be reset to 0, but "`i`" will remain unchanged. If an along-track update occurs, which will likely be the most common type of update (to compensate for wind prediction errors), then "`k`" will be incremented and "`i`" and "`j`" will remain unchanged. Along-track updates are discussed in the next section.

VI. Along-Track Updates

Airplanes normally cruise at constant airspeed (CAS or Mach), and the groundspeed corresponding to the most efficient airspeed obviously varies with the along-track wind speed. If the wind field prediction is accurate, then an efficient groundspeed can be determined. If the wind predictions are substantially in error, however, the airspeed corresponding to the assigned groundspeed could be inefficient or even unflyable. In other words, during periods when wind modeling and prediction is inaccurate, along-track conformance can be expensive or even impossible. The two relevant concerns here are the wind prediction accuracy and the speed range of the aircraft.

Reference[26] cites wind prediction accuracy results for the Rapid Update Cycle (RUC-1)[27,28] augmented with aircraft wind reports. The wind error vector magnitude exceeded 7.85 m/s (15.3 kn) only 10 percent of the time, and it exceeded 10 m/s (19.4 kn) only 4 percent of the time. The errors tend to be somewhat worse during the winter months because of higher wind speeds in general, but they are somewhat better than the quoted figures the rest of the year. The errors also tend to be larger at higher altitudes where the wind speeds are higher, but the figures quoted above are from actual commercial transport aircraft at their operating altitudes. These performance figures are likely to improve as research continues. Note that the along-track component of the wind error, which is the significant quantity here, is less than the vector magnitude. Averaged over all heading directions, the mean headwind error is $2/\pi(\approx 0.64)$ times the magnitude of the error vector (that overstates the average effective reduction, however, because neither heading nor wind directions are uniformly distributed).

The other relevant factor here is aircraft speed range. Figure 11 is a plot of the speed envelope for an MD-80 airplane at a gross weight of 135,000 lb on a standard day. This figure is taken from reference[29] and is based on data from a McDonnell Douglas performance handbook. These data, as well as the aircraft

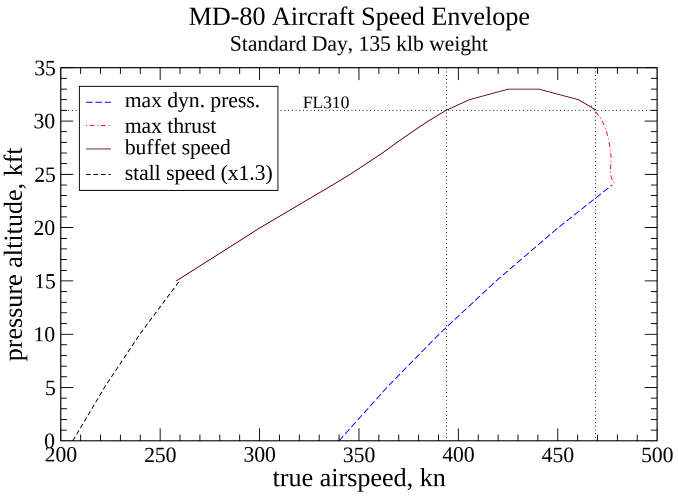

Figure 11. MD-80 Aircraft Speed Envelope for a standard day at 135 klb gross weight.

weight, could be known by the ground systems for all equipped aircraft. Any uncertainties in the weight would need to be accounted for to guarantee that the speed range is not overestimated.

Figure 11 shows that the speed range falls off sharply as the pressure altitude ceiling of approximately FL330 (for this weight) is approached. At FL310 the speed range is approximately 75 kn, and at FL290 it is slightly below 100 kn. Suppose the aircraft is flying at FL310 at the recommended cruise speed of Mach 0.76, which is equivalent to 446 kn at that altitude. The speed envelope then goes from approximately 394 to 469 kn, as shown in Figure 11, so the speed can be increased by a maximum of 23 kn or decreased by a maximum of 52 kn from the recommended speed. Thus, if the wind prediction error is within the range of -23 to $+52$ kn, the aircraft can maintain the assigned groundspeed exactly.

Although the aircraft can fly from 394 to 469 kn at FL310, it cannot fly efficiently over that entire range. Suppose the efficiency is deemed "acceptable" from 430 to 454 kn at that altitude and weight. Then, as long as the wind prediction error is within -8 to $+16$ kn, the aircraft can maintain the assigned groundspeed exactly and still fly with acceptable efficiency. Outside that range, an incentive exists to fly at an efficient airspeed until a point is reached at which the aircraft can no longer conform to the along-track error tolerances. Hence, additional rules may need to be established for how tightly an aircraft should track the assigned groundspeed, but such rules are beyond the scope of this paper.

The aircraft isn't required to fly the assigned groundspeed exactly; it is only required to stay within the along-track position bounds, which can vary linearly with time. Suppose the along-track tolerance rates are zero, the along-track position tolerances are ± 2 nmi, and the aircraft is centered within the along-track position bounds. If the along-track wind error is within ± 10 kn, the aircraft will be able to maintain its recommended airspeed for at least 12 min before it can possibly fall out of conformance.

Alternatively, if the along-track tolerance rates are ± 12 kn, then the along-track tolerances will increase at that rate, which is 0.2 nmi per minute, in each direction. Thus, the aircraft will be able to fly at the recommended airspeed and maintain a constant or increasing margin from the along-track bounds even if the along-track wind predictions are in error by up to ± 12 kn.

Commercial transport airplanes normally climb with throttle fixed somewhere in the range of 85 to 95 percent of full throttle, with feedback to the elevator to maintain constant CAS (at lower altitudes) or constant Mach (at higher altitudes). If the errors in the prediction of the along-track wind speed are reasonably small, the aircraft should still be able to fly in that mode. However, when altitude or along-track position approach their bounds, the throttle may also need to be adjusted to maintain conformance. The feedback to the throttle could be programmed to start automatically when the deviation reaches some threshold magnitude. The error tolerances should be set to accommodate the entire range of potential wind errors to some high level of certainty, say 99.9%. This could make the error tolerances fairly large, but at least they will clearly bound the area (as a function of time) that needs to be avoided by other aircraft. In the current system, the lack of explicit bounds forces controllers to reserve excessively large amounts of airspace for climbing and descending aircraft, which reduces airspace capacity.

In the event of substantial errors in the prediction of along-track winds, aircraft could be forced to fly at grossly inefficient speeds to maintain along-track position conformance. Worse yet, they could reach a state in which they are aerodynamically incapable of flying at the speed necessary to maintain conformance. To avoid either of those two undesirable conditions, particularly the latter, the along-track assignments can be updated periodically. The updates could apply to position, speed, and error tolerances. With proper updates, the worst that should happen is that some traffic may be forced to fly inefficiently for short periods of time to avoid a conflict, but they would obviously never be required to fly at speeds of which they are incapable of flying.

A complete 4D trajectory can (optionally) be filed by each participating aircraft or airline prior to takeoff. The trajectory reference time, which is specified by the "`reftime`" attribute of the "`trajectory`" element, will be defined as the time at which the aircraft is expected to cross some predefined marker, such as the end of the takeoff runway. Because takeoff time usually cannot be predicted exactly, the first along-track update will occur immediately after takeoff. When the aircraft crosses the reference marker, its reference time will be adjusted accordingly. Because all other times are relative to the reference time, no other times need to be changed. By adjusting the trajectory reference time, the entire trajectory can be effectively shifted in time.

After takeoff, the FMS will guide the aircraft along its assigned climb trajectory. As explained earlier, conventional feedback of speed error to the elevator will be used to maintain constant CAS or Mach, and feedback to the throttle will be used only if the vertical or along-track deviation reaches some threshold value, which shouldn't happen often if the wind predictions are reasonably accurate and the error tolerances are reasonable. If the aircraft does drift away from its reference trajectory and approach its along-track error bounds, however, the along-track assignment can be updated by changing the assigned position, speed, and/or error tolerances. Such updates would be allowed only if they do not cause a conflict within the conflict time horizon of, say, 15 min.

The most common type of along-track assignment update will likely be to change the assigned position to the current position and to simultaneously reset the position error tolerances to their initial values. The assigned groundspeed (the rate of change of the assigned along-track position) could also be changed if the wind model is determined to be significantly in error. This kind of update could be done periodically at a rate of, say, once per two minutes, except that it would *not* be done if it produces a potential conflict within the conflict time horizon. A conflict is defined as having the bounding spaces of two aircraft come closer together than the required minimum separation at the same time. In other words, conformance to their assigned trajectories by any two aircraft must guarantee the minimum required separation.

Because all the times given in the trajectory specification are relative to the trajectory reference time, the entire trajectory can be shifted in time by changing the reference time. That is equivalent to changing the assigned along-track position. As a flight progresses, trajectory segments that are completely in the past can be discarded. Because along-track updates are likely to be a common operation, an abridged format is appropriate, where unchanged data is not repeated. An example of a simple but complete along-track update is shown in XML Sample 10.

As explained earlier, the "`devtime`" attribute of "`flight`" gives the trajectory deviation time, which is the time at which the new trajectory deviates from the previous trajectory. The new trajectory reference time adjusts the trajectory to the current along-track position of the flight, and the along-track tolerances are reset to their nominal initial values. Note also that the last field of the trajectory revision number in the "`rev`" attribute is incremented for an along-track update. Although the entire trajectory specification is not transmitted, it will be stored on the ground and reconstructed onboard the aircraft based on the update and the previously stored trajectory.

Note that along-track updates can sometimes be used to resolve conflicts without maneuvering either aircraft. Suppose an aircraft has been flying for a relatively long time without an update, and its bounding space has grown to several miles in length in the along-track axis. All other traffic is required to maintain separation from that elongated bounding space. If a conflict then develops with a part of the bounding space that is sufficiently far away from the aircraft itself, an along-track update can be used to contract the bounding space around the aircraft, thereby removing the conflict without maneuvering either aircraft.

```
<flight ID="AAL2332/SFO" CID="324459"
    assigntime="14:05:32"
    devtime="14:06:32" rev="1.1.3"
    status="assigned">

    <trajectory reftime="14:02:17">
        <tolerances>
            <along tol="-2.0 2.0"
                rate="-10 10" time0="0:00"
                max="-10 10"/>
            </tolerances>
        </trajectory>

    </flight>
```

XML Sample 10. Along-track update

VII. Conclusion

An XML data format standard has been proposed for specifying trajectories for appropriately equipped aircraft in future high-capacity airspace. The format specifies a 4D reference trajectory along with error tolerances, which together define a bounding space, at each point in time, in which the aircraft is required to be contained. Trajectories can then be specified to guarantee the minimum required separation between any pair of conforming aircraft for a given period of time.

Trajectories are specified as a series of segments. The horizontal path consists of a series of greatcircle segments connected by turns of specified radius. Along-track position is specified as a low-order polynomial function of time, and vertical profiles for climb and descent are specified as low-order polynomial functions of along-track position. Error tolerances in the along-track, cross-track, and vertical axes determine the bounding space. Periodic updates in the along-track axis can be used to compensate for errors in the predicted along-track winds.

This regimen of 4D trajectories can eliminate the need for monitoring of equipped traffic by human air traffic controllers. It can also guarantee that the equipped traffic will be able to fly free of conflicts for at least several minutes even if all ground systems and the entire communication infrastructure fail. This failsafe guarantee, along with the elimination of the human factor from the primary separation feedback loop, has the potential to greatly increase airspace capacity.

References

[1] National Airspace System Operational Evolution Plan: A Foundation for Capacity Enhancement 2003-2013, Version 5.0, Executive Summary, Dec 2002. Available from http://www.faa.gov/programs/oep.

[2] Krozel, J.; Peters, M.; Bilimoria, K.; Lee, C.; and Mitchell, J. S. B: System Performance Characteristics of Centralized and Decentralized Air Traffic Separation Strategies. 4th USA/Europe Air Traffic Management R&D Seminar, Santa Fe, NM, USA, Dec 3-7, 2001.

[3] Andrews, J.: Capacity Benefits of the Automated Airspace Concept (AAC): A Preliminary Investigation. MIT Lincoln Laboratory Report No. 42PM-AATT-0014, Cambridge, MA, Aug. 2001.

[4] Erzberger, H.; and Pecsvaradi, T.: 4-D Guidance System Design with Application to STOL Air Traffic Control. (A72-38226 19-10), 13th Joint Automatic Control Conf., Stanford, CA, Aug 16-18, 1972, pp. 445-454.

[5] Wilson, I.: PHARE Advanced Tools Project Final Report. DOC 98-70-18, Eurocontrol, 21 Nov 1999.

[6] Sorensen, J., et al.: Massive Point-to-Point (PTP) and On-Demand Air Transportation System Concept Description. Volume I, Version 1.1. TR03231-01, NASA Contract No. NAS2-02076, Seagull Technology, Inc., Campbell, California, January 15, 2003.

[7] Rossow, V. J.: Use of Individual Flight Corridors to Avoid Vortex Wakes. *Journal of Aircraft*, vol. 40, no. 2, March-April 2003, pp. 225-231.

[8] Arkind, K.: Maximum Capacity Terminal Area Operations in 2022. AIAA Aviation Technology, Integration, and Operations (ATIO) Forum, Denver, CO, Nov 17-19, 2003.

[9] Erzberger, H.; and Paielli, R. A.: Concept for Next Generation Air Traffic Control System. Air Traffic Control Quarterly, vol. 10, no. 4, Winter 2002, pp. 355-378.

[10] Erzberger, H.: "Transforming the NAS: The Next Generation Air Traffic Control System," NASA/TP—2004-212828, Aug 2004.

[11] Paielli, R.A.; and Erzberger, H.: Tactical Conflict Detection Methods for Reducing Operational Errors, *Air Traffic Control Quarterly*, vol. 13, no. 1, 2005, pp. 83-106.

[12] Concepts for Services Integrating Flight Operations and Air Traffic Management Using Addressed Data Link, RTCA DO-269, June 12, 2001.

[13] Barrer, J. N.: Integrating the Cockpit with Air Traffic Management: The Concept of Path Objects. 3rd USA/Europe Air Traffic Management R&D Seminar, Napoli, Italy, June 13-16, 2000.

[14] Viets, K.J.; Taber, N.J.: An Overview of a Flight Object Concept for the National Airspace System, MITRE Technical Report MTR 00W0000085, Sept. 2000.

[15] Barhydt, R.; and Warren, A.: Development of an Information Structure for Reliable Communication of Airborne Intent and Aircraft Trajectory Prediction. *Air Traffic Control Quarterly*, vol. 11, no. 3, Fall 2003, pp. 251-274.

[16] Advanced Flight Management Computer System, ARINC Characteristic 702A-1, prepared by Airlines Electronic Engineering Committee, Jan 31, 2000.

[17] Minimum Aviation System Performance Standards for Automatic Dependent Surveillance Broadcast (ADS-B), RTCA DO-242A, RTCA, Inc., Washington, DC., 2002.

[18] Wilson, I.: PHARE: Definition and Use of Tubes. DOC 96-70-18, Eurocontrol, 15 July, 1996.

[19] Reynolds, T. G.; and Hansman, R. J.: Investigating Conformance Monitoring Issues in Air Traffic Control Using Fault Detection Approaches. ICAT-2003-5, Intl. Center for Air Transportation, Massachusetts Institute of Technology, Nov. 2003.

[20] Minimum Aviation System Performance Standards: Required Navigation Performance for Area Navigation, RTCA DO-236A, Sept. 13, 2000.

[21] See http://XML.org for general information on XML. See http://www.w3schools.com/xml for a tutorial on XML.

[22] Denery, D. G.; and Erzberger, H.: The Center-TRACON Automation System: Simulation and Field Testing. NASA/TM—1995-110366. See also http://www.ctas.arc.nasa.gov/

[23] Slattery, R.; and Zhao, Y.: Trajectory Synthesis for Air Traffic Automation. *J. Guidance, Control, and Dynamics*, vol. 20, no. 2, March-April, 1997, pp. 232-238.

[24] Lindsay, K. S.; Green, S. M.; Mondoloni, S.; and Paglione, M.: Common Trajectory Modeling for National Airspace System: Decision Support Tools. *Air Traffic Control Quarterly*, vol. 13, no. 1, 2005, pp. 55-82.

[25] Paielli, R. A.: A Linear Altitude Rule for Safer and More Efficient Enroute Air Traffic. *Air Traffic Control Quarterly*, vol. 8, no. 3, Fall 2000, pp. 195-221.

[26] Cole, R. E.; Green, S.; Jardin, M.; Schwartz, B. E.; and Benjamin, S. G.: Wind Prediction Accuracy for Air Traffic Management Decision Support Tools. *3rd USA/Europe Air Traffic Management R&D Seminar*, Napoli, Italy, 13-16 June 2000.

[27] Benjamin, S. G.; J. M. Brown; K. J. Brundage; D. Devenyi; G. A. Grell; D. Kim; B. E. Schwartz; T. G. Smirnova; T. L. Smith; S. Weygandt; and G. S. Manikin: RUC20 - The 20-km version of the Rapid Update Cycle. NOAA Technical Memorandum OAR FSL-28, 2002 (available from http://maps.fsl.noaa.gov).

[28] Cole, R. E.; Richard, C.; Kim, S.; and Bailey, D.: An Assessment of the 60 km Rapid Update Cycle (RUC) with Near Real-Time Aircraft Reports. NASA/A-1, Lincoln Laboratory, 15 July 1998.

[29] Mueller, K. T.; Schleicher, D. R.; and Bilimoria, K. D.: Conflict Detection and Resolution with Traffic Flow Constraints. *AIAA Guidance, Navigation, and Control Conf.*, Monterey, California, Aug 5-8, 2002.

About the Author

Russ Paielli received a B.S. degree in Mechanical Engineering from Oakland University in Michigan in 1982 and an M.S. degree in Aeronautics and Astronautics from Stanford University in 1987. He has worked at NASA Ames Research Center since 1982. At Ames he has done research in flight control theory and precision aircraft navigation and landing. He is currently working on advanced concepts in Air Traffic Management, with a focus on separation assurance. He is an Associate Fellow of AIAA. He maintains a personal website at http://RussP.org. [paper last revised on 2006-09-19]

A. Greatcircle Formulas

The earth is nearly but not quite spherical, with an eccentricity of approximately one part in 300. Greatcircle algorithms come in two forms: those based on a simplified spherical model of the earth, and those based on a more accurate but more complex ellipsoidal model. The spherical model yields closed-form analytic solutions, whereas the ellipsiodal model yields more accurate but more complicated iterative algorithms. The ellipsoidal greatcircle algorithms may be needed for accurate along-track distances for very long segments, but they are too complicated to be presented here. A few of the key spherical greatcircle equations are presented here for reference.

The geodetic coordinates of a point A on the surface of the earth will be represented as (ϕ_A, λ_A), where ϕ_A is the latitude, and λ_A is the longitude. Given a greatcircle starting from point A with initial course angle χ_A, the point a distance d (in radians) away from point A along the greatcircle at (ϕ, λ) is given by

$$\begin{aligned} \phi &= \operatorname{asin}(\sin\phi_A \cos d + \cos\phi_A \sin d \cos\chi_A) \\ \lambda &= \operatorname{mod}(\lambda_A - \operatorname{atan2}(\gamma_1, \gamma_2) + \pi, 2\pi) - \pi \end{aligned} \tag{1}$$

where

$$\begin{aligned} \gamma_1 &\equiv \sin\chi_A \sin d \cos\phi_A \\ \gamma_2 &\equiv \cos d - \sin\phi_A \sin\phi \end{aligned}$$

Given points A and B on the surface of the earth at (ϕ_A, λ_A) and (ϕ_B, λ_B), the angular distance between them is given by

$$\text{distance}(A, B) = 2\ \operatorname{asin}(\gamma_1^2 + \cos\phi_A \cos\phi_B \gamma_2^2) \tag{2}$$

where

$$\begin{aligned} \gamma_1 &\equiv \sin((\phi_B - \phi_A)/2) \\ \gamma_2 &\equiv \sin((\lambda_A - \lambda_B)/2) \end{aligned}$$

The angular distance must be multiplied by the radius of the earth to determine the actual length of the greatcircle. The official FAA earth radius is 3440.655273 nmi.

Given a greatcircle from point A at (ϕ_A, λ_A) to point B at (ϕ_B, λ_B), the initial course angle of the greatcircle at point A is given by

$$\text{course}(A, B) = \operatorname{atan2}(\beta_1, \beta_2) \tag{3}$$

where

$$\begin{aligned} \beta_1 &\equiv \sin(\lambda_A - \lambda_B) \cos\phi_B \\ \beta_2 &\equiv \cos\phi_A \sin\phi_B - \sin\phi_A \cos\phi_B \cos(\lambda_A - \lambda_B) \end{aligned}$$

Note that the course angle at the *end* of the greatcircle can be determined by simply reversing the order of the arguments of the course function, which is useful for determining the turn angle between consecutive greatcircle segments.

Given these utility functions, the cross-track error (positive to the right) of a point C that was supposed to be on the greatcircle from A to B is given by

$$\begin{gathered} \text{cross}(A, B, C) = \operatorname{asin}(\sin(A, C) \\ \sin(\text{course}(A, C) - \text{course}(A, B))) \end{gathered} \tag{4}$$

where

$$\sin(A, C) \equiv \sin(\text{distance}(A, C))$$

The along-track position is given by

$$\begin{gathered} \text{along}(A, B, C) = \operatorname{asin}(((\sin(A, C))^2 - \\ (\sin(\text{cross}(A, B, C)))^2)^{1/2} / \cos(\text{cross}(A, B, C))) \end{gathered} \tag{5}$$

As before, these results must be multiplied by the radius of the earth to convert them from angles to lengths.