Point-Mass Aircraft Trajectory Prediction Using a Hierarchical, Highly-Adaptable Software Design

David A. Karr,* Robert A. Vivona,† Sharon E. Woods‡

Engility Corporation, Billerica, MA 01821, USA

David J. Wing§

NASA Langley Research Center, Hampton, VA 23681, USA

A highly adaptable and extensible method for predicting four-dimensional trajectories of civil aircraft has been developed. This method, Behavior-Based Trajectory Prediction, is based on taxonomic concepts developed for the description and comparison of trajectory prediction software. A hierarchical approach to the "behavioral" layer of a point-mass model of aircraft flight, a clear separation between the "behavioral" and "mathematical" layers of the model, and an abstraction of the methods of integrating differential equations in the "mathematical" layer have been demonstrated to support aircraft models of different types (in particular, turbojet vs. turboprop aircraft) using performance models at different levels of detail and in different formats, and promise to be easily extensible to other aircraft types and sources of data. The resulting trajectories predict location, altitude, lateral and vertical speeds, and fuel consumption along the flight path of the subject aircraft accurately and quickly, accounting for local conditions of wind and outside air temperature. The Behavior-Based Trajectory Prediction concept was implemented in NASA's Traffic Aware Planner (TAP) flight-optimizing cockpit software application.

I. Introduction

PREDICTING the four-dimensional trajectory of an aircraft, including lateral path, altitudes, longitudinal and vertical speeds, and fuel consumption, is an important tool to support flight planning and air traffic management applications. Experience has shown that the development of software for making high-quality predictions of aircraft trajectories is fraught with challenges, especially in the face of changing requirements regarding outcomes and data sources. The Behavior-Based Trajectory Prediction software concept aims to give developers a powerful, highly-adaptable framework for accurate, efficient, self-documenting, and easily-maintained trajectory-prediction software.

The remainder of this paper is organized as follows. The second section lays out the motivation for developing an implementation of Behavior-Based Trajectory Prediction and the prior lessons learned that led to the selection of this concept for the development of a new software entity. The third section describes the implementation of the layers of trajectory modeling, giving an example of how Behavior-Based Trajectory Prediction enables modeling the performance of different types of aircraft using different data sources. The final sections of this paper discuss the results of the current effort and anticipated future work.

[*] Chief Technical Specialist, ATRE Department, 300 Concord Road, AIAA senior member.

[†] Chief Research Engineer, ATRE Department, 300 Concord Road, AIAA associate fellow.

[‡] Principal Software Engineer, ATRE Department, 300 Concord Road, AIAA member.

[§] Principal ATM Research Engineer, Crew Systems & Aviation Operations, M.S. 152, AIAA senior member.

II. Motivation

We developed the Behavior-Based Trajectory Prediction concept for use in NASA's Traffic Aware Strategic Aircrew Requests (TASAR) project.[1] TASAR utilizes a cockpit-based software tool, the Traffic Aware Planner (TAP),[2] as an aid to identifying in-flight route-optimization opportunities that—by also avoiding nearby air traffic, weather, special-use airspace (SUA), and other constraints—are more likely to be approved by Air Traffic Control (ATC) when requested by the pilot. This environment imposed several requirements on the part of the software implementation that would predict trajectories.

In order to produce and display a proposed route optimization, TAP evaluates numerous candidate *aircraft intents,* each consisting of a route with speed and altitude settings. To ensure the suitability of each candidate, TAP uses a model of the aircraft's performance to predict the full four-dimensional trajectory that the aircraft would follow from its current *aircraft state* according to the aircraft intent in question, and checks the trajectory for conflicts with other aircraft, with weather cells, and with SUAs. Furthermore, in order to optimize the routes, TAP also predicts fuel consumption and time of arrival at the destination. For example, to evaluate how much fuel might be saved if the crew were to increase the cruise altitude 4000 feet for the duration of the flight before landing, provided that the aircraft can fly safely at that altitude, TAP accounts for the additional fuel required to climb to the new cruise altitude as well as the savings incurred by cruising at that altitude and by lengthening the descent.

In order to adequately consider potential optimizations that might change the lateral path of the route, the cruise altitude, or both, TAP must be able to predict the trajectories of hundreds of candidate aircraft intents in at most a few seconds of real time. Moreover, TAP is required to do all of this while using limited computational resources such as are available on an electronic flight bag (EFB) shared with other applications.

At the time development of TAP began, we had already planned to deploy TAP on a Piaggio Avanti turboprop aircraft, in addition to commercial turbojet aircraft. TAP was able to re-use much of the software earlier developed for NASA's Autonomous Operations Planner (AOP),[3] a cockpit automation system for separation assurance. We soon determined, however, that the type of performance data available for the Piaggio Avanti required a fundamentally different approach to trajectory prediction than the one previously used in AOP. In addition, it was desirable to be able to use the most accurate performance data that could be obtained for the specific types of turbojet aircraft on which deployments might be made rather than approximating them with data for similar aircraft in the database of aircraft models used by AOP. The requirements of TAP would include the ability to use models of various aircraft whose available performance data not only had different numeric values but were encoded in different ways or that described different aspects of performance, avoiding limitations on the types and sources of data that could be used.

In summary, TAP required trajectory prediction software that would meet the following requirements:

1. It needed to be adaptable to various aircraft with disparate types of performance.
2. It needed to populate its various aircraft models with disparate types of performance data presented in disparate encodings and formats.
3. It needed to provide precise and accurate predictions of lateral position, time, and altitude along the trajectory as well as total fuel consumption.
4. It needed to complete predictions quickly in real time.
5. It needed to enable trade-offs among the cost of execution of the software, the availability of suitable performance data, and the fidelity of the trajectory predictions.

Due to these requirements, we decided to design and implement a new software entity for trajectory prediction rather than re-use or adapt existing software for this purpose. The resulting software is the Behavior-Based Trajectory Prediction software module implemented in TAP.

III. Design and Implementation

A. Conceptual Plan

Design of the new trajectory prediction software drew heavily from lessons learned from previous experience with trajectory prediction. In order to have a clear conceptual basis for the algorithms and to maximize the ability to adapt the software to new types of aircraft and performance models, we decided that the design of the new trajectory-prediction software would be based on abstraction techniques that were developed in support of the United States-Europe Air Traffic Management Research and Development Action Plan 16 (AP16) for Common Trajectory Prediction Capabilities.[4] In particular, AP16 distinguishes behavioral modeling (descriptions of how the crew or guidance systems of the aircraft carry out the intent of the aircraft's flight plan, including constraints to be met) from mathematical modeling (the equations of motion and the methods for integrating those equations along the trajectory).

We applied the behavioral-mathematical abstraction literally in the new software design: the initial aircraft state and intent from which a trajectory is to be predicted are passed to a layer of the software consisting entirely of *behavior models* instantiated as C++ objects; in order to numerically predict the instantaneous states of the aircraft (latitude, longitude, altitude, speed, weight, and so forth) at all points along the trajectory, the behavior models call upon a separate layer of the software consisting of *math models* (instantiated as distinct C++ objects) to perform the necessary calculations. We call the implementation of this design Behavior-Based Trajectory Prediction because of the explicit separation of the behavioral layer of abstraction from the underlying mathematical models.

B. The Behavioral Layer

The behavioral layer of Behavior-Based Trajectory Prediction is itself organized in a hierarchy of behavior models. At the top of the hierarchy is a single object, the BehaviorBasedTrajectoryBuilder. Immediately under this object are a LateralBehaviorModel and a VerticalBehaviorModel, which are responsible respectively for defining how the aircraft will follow a lateral path over the surface of the Earth and for defining whether and how the aircraft will climb or descend along that path. The VerticalBehaviorModel is also responsible for defining how the speed of the aircraft along the trajectory will be controlled.

The types LateralBehaviorModel and VerticalBehaviorModel are abstract classes to be instantiated by concrete derived classes. For the TAP application, the only type of intent that needed to be supported was the intent to follow a lateral route at a given cruise altitude, possibly with constraints at some intermediate waypoints, ending at a destination airfield. Initially, therefore, only one top-level lateral behavior model (named LnavBehaviorModel) and one top-level vertical behavior model (named VnavVerticalBehaviorModel) were developed. Future versions of the software, however, may substitute alternative behavior models for some trajectories in order to predict aircraft intents such as "turn to heading" or "hold altitude." This would enable Behavior-Based Trajectory Prediction to support applications such as NASA's AOP (see Section II), which provides a variety of tactical maneuvers based on such aircraft intents (in addition to strategic maneuvers that follow routes with constraint waypoints) in order to meet the requirements of airborne self-separation.

C. Lateral Modeling

The route-following lateral behavior is composed of a sequence of models of sub-behaviors that form the next layer of the hierarchy under the main lateral behavior. The sub-behaviors include flying "straight" along the legs of the route between waypoints, fly-by turns at waypoints whose turn angles exceed a threshold value, a "single point" behavior at the end of the route, and an optional initial turn that can be inserted as the first sub-behavior model when the initial state for trajectory prediction is on the arc of a turn. These sub-behaviors are *atomic behaviors*, that is, they are not themselves composed of sub-behavior models.

Each atomic lateral behavior has a single lateral math model. The lateral math models keep track of the predicted start and end of each turn and each "straight" portion of the trajectory. Any lateral position on the path can be identified by its atomic behavior model and a distance from the reference point of that model, measured in the forward direction along the lateral path. Moreover, each lateral math model has settings for the position of the start and (optionally) the end of the portion of the trajectory that is located within that model. The positions of the start and end of the lateral math model are measured along the path in the forward direction. A negative position value indicates that the point occurs before the reference point of the model.

The math model of a fly-by turn waypoint behavior has its start and end at the points of tangency of the turn arc with the legs to and from the waypoint, as shown in Figure 1. (The direction of travel is from left to right.) The start and end points of the great-circle math model within a route-leg behavior model are set to the end of the turn (if any) at the waypoint at the start of the leg and the start of the turn (if any) at the waypoint at the end of the leg. This often causes the math model to start after its reference point, as shown in Figure 2.

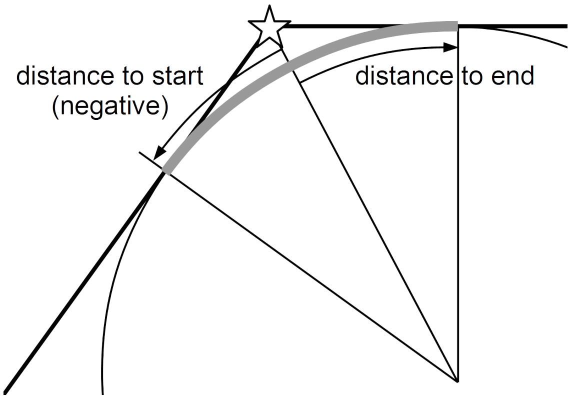

Figure 1. Geometry of a small-circle math model in a fly-by turn waypoint behavior model.

Except for the optional initial turn (which is assumed to be a turn onto the next leg of the route), the radius of each turn in the route is initially set to zero. The final (non-zero) radius is applied under the control of a vertical behavior model, which has the ability to estimate the groundspeed in the turn.

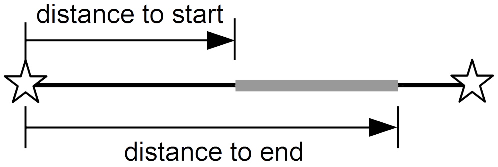

Figure 2. Geometry of a great-circle math model in a route leg behavior model.

The decision to model turn arcs in routes at all is an example of the flexibility and adaptability of the Behavior-Based Trajectory Prediction concept. Neither the concept nor its implementation prevent the construction of a route-following lateral behavior that plots courses directly to and from waypoints without modeling turn arcs between legs. In such a design, the lateral behavior models and lateral math models for turns would simply not be instantiated at run time. Such a design would trade off some of the fidelity of the predictions used in TAP in return for lateral models that were much simpler, allowing faster execution of trajectory prediction.

D. Vertical Modeling

In contrast to lateral behavior modeling, we found it unsuitable for TAP to try to construct the full sequence of vertical behavior models simply by scanning the intent in the input to trajectory prediction. In the development of TAP, vertical behavior models tended to have deeper and more complex hierarchies of sub-behaviors than lateral behavior models, and we found it best to postpone the construction of vertical sub-behaviors until a sufficient part of the trajectory had been predicted that it was possible to decide whether a particular sub-behavior is necessary.

The main vertical behavior model used in TAP predictions contains up to three models of sub-behaviors. If the aircraft is already in descent, the main vertical behavior includes only a "descent" sub-behavior; if the aircraft is cruising, the main vertical behavior includes a "cruise" sub-behavior followed by a descent;

otherwise, the main vertical behavior model is composed of a "climb" behavior model followed by cruise and descent models.

The main vertical behavior model performs the following sequence: (1) it integrates the climb sub-behavior (when present) forward from the initial state to the top of climb; (2) it uses the first predicted state in cruise to estimate the weight at destination with the help of a suitable math model; (3) it integrates the descent sub-behavior backward from the destination (using the estimated weight at that point) to the top of descent; (4) it integrates the cruise sub-behavior from the top of climb to the top of descent; and (5) it uses the difference between the two predicted weights at the top of descent (one from the cruise sub-behavior, one from the descent sub-behavior) to correct the predicted weights during the descent.

Within every other vertical behavior model yet implemented for TAP, the sub-behaviors are evaluated in sequence so that either the math models are integrated forward along the trajectory from the start of the high-level behavior toward the end (if the behavior occurs within the climb or cruise behaviors) or the math models are integrated backward from the end of the high-level behavior (if the behavior occurs in descent). The lateral position, altitude, speeds, and other data predicted at the boundary of one sub-behavior become part of the input to the prediction of the sub-behavior on the other side of the boundary.

Four key elements of the Behavior-Based Trajectory Prediction methodology are described next: hierarchies of behavior models, equations of motion, numerical integration, and aircraft performance models.

1. Hierarchies of Behavior Models

Each behavior within a trajectory typically is composed of its own unique hierarchy of types of behavior models. Let us consider climbing to the initial cruise altitude as an example. Table 1 (adapted from Table 2 in Ref. 4) shows how the AP16 effort applied its behavioral-mathematical abstraction to this part of a flight.

Table 1. Behavioral and Mathematical Models for Example Climb Maneuver for AP16.

Behavioral model		Mathematical model					
Maneuver	**Sub-maneuver**	**Equations of motion**	**Speed**	**Aircraft performance model**	**Integration direction**	**Integration algorithm**	**Time step**
CLIMB	Indicated Airspeed (IAS) CLIMB	point-mass: known speed and thrust	constant IAS	kinetic; thrust from Max Climb table	forward	2nd-order Runge-Kutta	60 seconds
	MACH CLIMB	point-mass: known speed and thrust	constant Mach	kinetic; thrust from Max Climb table	forward	2nd-order Runge-Kutta	60 seconds

For comparison, the hierarchy of models in the TAP Behavior-Based Trajectory Prediction implementation for a civil turbojet transport aircraft climbing to its cruise altitude without climb constraints is shown in Table 2. This table is very similar to Table 1, with a few significant differences.

First, because Behavior-Based Trajectory Prediction maps behaviors directly to hierarchical objects in its implementation, each "maneuver" called out in Table 1 is implemented as a separate behavior model; hence we use the term "behavior" rather than "maneuver" to label the columns in Table 2.

Second, Behavior-Based Trajectory Prediction has an additional layer of behavior models in its hierarchy in this example: whereas the atomic behavior model "IAS climb max thrust" (where IAS is indicated airspeed) has an obvious correspondence to the IAS CLIMB sub-maneuver in the AP16 model, the implementation of Behavior-Based Trajectory Prediction inserts the behavior "climb below IAS/Mach transition"

Table 2. Behavioral and Mathematical Models for Turbojet Climb in TAP.

Behavioral model			Mathematical model					
Behavior	**Sub-behavior**	**Atomic behavior**	**Equations of motion**	**Speed**	**Aircraft performance model**	**Integration direction**	**Integration method**	**Integration step**
climb	climb below IAS/Mach transition	IAS climb max thrust	point-mass; known speed and thrust setting	constant IAS	kinetic; thrust, drag from state functions	forward	2nd-order Runge-Kutta (midpoint method)	2000 feet
	climb above IAS/Mach transition	Mach climb max thrust	point-mass; known speed and thrust setting	constant Mach	kinetic; thrust, drag from state functions	forward	2nd-order Runge-Kutta (midpoint method)	2000 feet

above the atomic model. If the IAS climb max thrust behavior had to be interrupted to avoid a constraint violation, there would be additional atomic behaviors within the "climb below IAS/Mach transition" behavior. But even when only one atomic model is needed, the extra layer provides a chance to identify the purpose of that part of the trajectory as opposed to "how" the aircraft is being flown.

This example is not meant to imply that Behavior-Based Trajectory Prediction always uses this same number of layers of behavior models along all parts of a trajectory. By design, any VerticalBehaviorModel can be a member of any non-atomic VerticalBehaviorModel. In practice, Behavior-Based Trajectory Prediction typically uses varying numbers of layers for different parts of a trajectory.

A third noteworthy difference between the AP16 and TAP models is that rather than requiring use of a time step, the math models in Behavior-Based Trajectory Prediction can accept various types of integration steps. This is a significant feature of numeric integration in Behavior-Based Trajectory Prediction that allows the "climb" behavior to integrate the climb in increments of 2000 feet instead of 60 seconds.

The "climb" behavior model for the Piaggio Avanti turboprop aircraft model used in the TAP flight tests is a significant departure from the AP16 example. This turboprop behavior model is shown in Table 3.

Table 3. Behavioral and Mathematical Models for Turboprop Climb in TAP.

Behavioral model			Mathematical model					
Behavior	**Sub-behavior**	**Atomic behavior**	**Equations of motion**	**Speed**	**Aircraft performance model**	**Integration direction**	**Integration method**	**Integration step**
climb	climb across speed transition	IAS schedule climb max power	point-mass; known speed and power setting	IAS table lookup	kinematic; vertical rate from tables	forward	2nd-order Runge-Kutta (midpoint method)	2000 feet

Similarly to a turbojet aircraft, the Piaggio Avanti climbs at a constant IAS only up to a certain altitude, after which IAS decreases with increasing altitude. Unlike the turbojet model, however, the transition altitude is predetermined and IAS decreases linearly above that altitude. This implied that IAS is a piecewise linear continuous function of altitude. Since the vertical math models of Behavior-Based Trajectory Prediction already supported such functions via lookup tables, a single math model was able to compute the entire climb.

Alternatively, using math models based on simple mathematical functions, the turboprop climb behavior could be divided into two sub-behaviors, one below the transition from constant IAS to linearly decreasing IAS and one above the transition. Table 4 shows how such a model could be defined.

Table 4. Alternative Behavioral and Mathematical Models for Turboprop Climb in TAP.

Behavioral model			Mathematical model					
Behavior	**Sub-behavior**	**Atomic behavior**	**Equations of motion**	**Speed**	**Aircraft performance model**	**Integration direction**	**Integration method**	**Integration step**
climb	climb below speed transition	IAS climb max power	point-mass; known speed and power setting	constant IAS	kinematic; vertical rate from tables	forward	2nd-order Runge-Kutta (midpoint method)	2000 feet
	climb above speed transition	IAS schedule climb max power	point-mass; known speed and power setting	IAS linear function	kinematic; vertical rate from tables	forward	2nd-order Runge-Kutta (midpoint method)	2000 feet

Like the turbojet models, the “climb” model for the Avanti integrates forward in steps of 2000 feet using second-order Runge-Kutta numeric integration. Unlike the turbojet models, however, the turboprop’s behavior model specifies maximum power rather than maximum thrust, and the vertical rate is determined by a function of several variables implemented via lookup tables. Although weight is one of the input variables of the vertical rate function, there is no explicit calculation of acceleration due to forces on the aircraft (that is, thrust, lift, and drag are not mentioned in the performance data), and therefore the model is considered kinematic rather than kinetic.

2. *Equations of Motion*

Each vertical math model contains an object that implements a sufficient set of equations to predict the relationships between distance traveled, altitude, weight, and time over its portion of the predicted trajectory. The actual type of object used may be any of several concrete classes derived from the abstract base class VerticalEquations. In some cases these equations are easily specified as a composition of three objects whose abstract base class types are named SpeedEquation, AltitudeEquation, and WeightEquation, by which the VerticalEquations object computes respectively the groundspeed, vertical speed or altitude, and fuel consumption over one step of a numeric integration. It soon became evident, however, that some other equations of motion are better modeled by composing vertical rate, drag, and fuel flow, while some other equations require thrust, drag, and fuel flow. Each way of composing the equations of motion (including equations for level flight) is implemented by a different concrete class derived from VerticalEquations.

3. *Numerical Integration*

The math model also has a reference to an object of the abstract class VerticalIntegrationMethod. The concrete class that instantiates this object determines which numerical integration rule is applied. For TAP, only two integration rules have been implemented so far. Both rules are types of second-order Runge-Kutta integration: the explicit midpoint rule, and the explicit trapezoid rule (Heun’s method). The interface to VerticalIntegrationMethod, however, is extremely simple and could easily support either higher-order or lower-order rules.

The variables that are integrated by the equations of motion may include elapsed time, distance along the path, altitude, the gross weight of the aircraft, indicated airspeed, Mach number, true airspeed, groundspeed, and vertical rate. Numerical integration results in vertical profile, which is a sequence of “points” consisting of these variables as well as others (such as temperature, pressure, or wind speed and direction) that may vary as the trajectory passes through different lateral positions and altitudes.

In order to minimize processing time, many of the variables along the trajectory are computed only when they are first required for use in the equations of motion. Once it has been computed, the value of the variable is usually cached so that it can be used multiple times in a set of equations with only a minimal added cost in processing time.

4. Independent Variables of Integration

In a typical trajectory-prediction algorithm using numeric integration, elapsed time is the independent variable on which all other variables are assumed to depend. The mathematical theory that justifies this approach, however, does not explicitly specify what variable must be used as the independent variable; it is theoretically possible to choose any variable such that the other variables in question can be treated as differentiable functions of the independent variable. Time is a “safe” choice of independent variable for trajectory prediction because the aircraft can never “stop time” or “fly back” to an earlier instant in time.

A disadvantage of using time as the independent variable is that the events along a trajectory at regular time intervals are usually not the events that matter. For example, if the aircraft is predicted to be at 31,300 feet at some point in time, climbing at 2180 feet per minute, we have no use for information about the point the aircraft will reach if it continues climbing for another 60 seconds, except to the extent that once we have predicting where the aircraft would be after 60 seconds of uninterrupted climb, we can interpolate when and where the aircraft will reach 32,000 feet and stop climbing. A more direct solution would be to integrate directly to flight level 320, if we can. Furthermore, the variations in pressure and temperature that tend to cause much of the variation in groundspeed are functions of altitude rather than time, which suggests that the difference in altitudes over a time step is the main determiner of accuracy in trajectory integration even below the cruise altitude.

A similar concern arises when we predict level flight over a route: we want to know when the aircraft will arrive at the next turning point, not where it would be if it flew straight for three or four minutes past that point. Even within the “straight” portions of the route, the local winds tend to vary more with distance along the path than with time (at least in the context in which TAP is currently planned to be used), so the distance covered in a time step may be a better indication of accuracy than the elapsed time would be.

These arguments do not show that time is a poor choice of an independent variable; they merely lead us to consider that time may not *always* be our preferred choice. Experience with Behavior-Based Trajectory Prediction has shown that for the level-flight portions of a trajectory in TAP, in which the aircraft is always predicted to fly forward along a path at a positive groundspeed, either time or distance flown along the path may be used as an independent variable. For portions of a trajectory that contain only a planned continuous climb or a planned continuous descent from one altitude to another, either time, distance flown, or altitude may be used as an independent variable.

Within these portions of the flight, equations that are based on derivatives of their variables with respect to time can be converted to use derivatives with respect to altitude (or to path length, in non-level flight) via the Chain Rule. Using t to represent time, s to represent distance along the path, h for pressure altitude, and y for any other property of the aircraft that can be expressed as a differentiable function of time (so that dy/dt is defined), if the groundspeed, V_g, is always positive,

$$\frac{dy}{ds} = \frac{dy}{dt} \cdot \frac{dt}{dh} = \frac{1}{V_g} \cdot \frac{dy}{dt},$$

whereas if the vertical speed, dh/dt, is always positive (alternatively, if dh/dt is always negative),

$$\frac{dy}{dh} = \frac{dy}{dt} \cdot \frac{dt}{dh} = \frac{1}{dh/dt} \cdot \frac{dy}{dt}.$$

As an example, suppose one of the behaviors in the trajectory is to climb at Mach 0.780 (held constant) with thrust at the MAX CLIMB setting. Given a weather prediction including wind and temperature throughout the surrounding airspace, we can determine the local conditions surrounding any latitude, longitude, and altitude through which the aircraft might pass. Suppose we use this knowledge to compute the aircraft's true airspeed, V_T, the rate of change of V_T with altitude, dV_T/dh, and the rate of change of tailwind with altitude, dU_W/dh, in any predicted aircraft state. Moreover we use an aircraft performance model to determine the drag, D, thrust, T, and rate of change of weight, $\dot{W}$ (equal to the fuel flow), under these conditions. We can then use the total-energy formula

$$\frac{dh}{dt} = \frac{T - D}{\frac{W}{V_T} + \frac{W}{g}\left(\frac{dV_T}{dh} + \frac{dU_W}{dh}\right)}$$

to find the vertical speed. The reciprocal of the vertical speed is therefore

$$\frac{dt}{dh} = \frac{\frac{W}{V_T} + \frac{W}{g}\left(\frac{dV_T}{dh} + \frac{dU_W}{dh}\right)}{T - D}.$$

Provided that we use this method only when the climb maintains at least a certain minimum vertical rate,[a] the derivative dt/dh is bounded and fairly well-behaved. Measuring altitudes in feet, we can use a standard numerical method for differential equations (such as Runge-Kutta) to integrate dt/dh from $h = 30000$ to $h = 32000$, and thereby find the amount of time that elapses while climbing from FL300 to FL320. Similarly, we can integrate the rate of change of weight with respect to altitude,

$$\frac{W}{dh} = \frac{W}{dt} \cdot \frac{dt}{dh} = \dot{W} \cdot \frac{dt}{dh},$$

over the same interval to determine the amount of fuel consumed during that portion of the climb. Finally, using a formula to compute the groundspeed, V_g, from the aircraft state and the wind at any given point, and letting s be the distance traveled along the lateral path, we have

$$\frac{ds}{dh} = \frac{ds}{dt} \cdot \frac{dt}{dh} = V_g \cdot \frac{dt}{dh},$$

which we can integrate over the same interval of altitudes in order to find the distance traveled during this portion of the climb.

It is possible to use altitude as an independent variable within behaviors where the aircraft is guaranteed to be always climbing (or alternatively, always descending) because each behavior is predicted by a separate math model with its own equations of motion, integrated independently from every other behavior except possibly for the sharing of an instantaneous state at a single time, altitude, and path position on the boundary between the two behaviors. The math model for each atomic behavior can therefore use an independent variable different from that used by the math model for another part of the trajectory.

The independent variable of a math model must nevertheless be suitable for the types of motion that may occur within the same behavior. If a behavior cannot guarantee altitude will be strictly increasing (or strictly decreasing) as a function of time, time or distance should be used as the independent variable instead of altitude. If distance along the path is not strictly increasing as a function of time, then time should be used as the independent variable instead of distance. For example, a hovering behavior could be implemented within Behavior-Based Trajectory Prediction by integrating over time.

[a] The algorithms implemented in Behavior-Based Trajectory Prediction include safeguards to prevent integrating over an altitude interval if the vertical rate would fall below a threshold or if the elapsed time or lateral distance would be excessive.

5. *Aircraft Performance Models*

A set of equations of motion often requires the evaluation of some functions of aircraft state and/or local atmospheric conditions. To support this, Behavior-Based Trajectory Prediction provides various objects that can be assembled into evaluators of the necessary functions. These include objects that extract the values of integrated variable from a point in a vertical profile, objects that compute standard functions of those data, and objects that evaluate functions of one or more numeric variables according to the performance model of the aircraft.

While a particular function within a particular aircraft performance model can be calculated by a custom-built C++ class if necessary, in many cases the class responsible for that function will use one of several generic types of *function evaluator* to perform the calculation. The types of calculation already supported in this way include interpolation by various methods within numeric tables of up to four dimensions, either completely regular ("rectangular") or with varying sizes in one or two dimensions, and polynomials in one or two variables. The function evaluator obtains the contents of its table or the coefficients of its polynomial from the aircraft performance model. Function evaluators are designed to be easy to compose into higher-dimensional function evaluators when required.

E. Coordination of Lateral and Vertical Behaviors

The output of the integration of each vertical math model is a vertical profile that maps distance along the path to the other variables predicted by vertical math models. The math model is provided with a reference to the lateral behavior, by means of which it can determine the lateral position (latitude and longitude) and course angle at any given distance along the path. These data enable the vertical math model to probe a model of current winds and air temperatures at each location along the trajectory, which in turn enables it to compute formulas that depend on local atmospheric conditions and to convert between true airspeed and groundspeed.

After it has predicted all of its vertical behaviors, Behavior-Based Trajectory Prediction traverses all lateral behaviors and vertical behaviors in parallel, starting at the beginning of the trajectory. The result of this procedure is a sequence of TrajectoryPoint objects that combine the vertical profile of each vertical math model with information from the lateral models to describe the instantaneous state of the aircraft at each significant point along the trajectory.

IV. Results of Current Effort

The development of Behavior-Based Trajectory Prediction is ongoing, but several milestones in implementation and deployment are worth noting.

A. Deployment aboard a Turboprop Aircraft

The initial implementation of Behavior-Based Trajectory Prediction was used in the second flight test of TAP in June 2015, in which a fully-functioning version of TAP was installed on an EFB connected to the avionics aboard a Piaggio Avanti turboprop aircraft.[2]

This implementation included behavior models for a turboprop aircraft, math models required to predict these behaviors, performance data for the Piaggio Avanti of the types available from a Pilot's Operating Handbook (formatted as numeric tables), suitable classes of objects to evaluate the required state functions by interpolating within the tables of performance data, and all the parts of the general design required to predict trajectories.

B. Simulation of Turbojet Aircraft

After the second flight test of TAP, we developed behavior models and math models suitable for a turbojet transport aircraft. These models read performance data from numeric tables available at the NASA Langley Research Center. These tables provide thrust, drag, fuel flow, and other data for turbojet aircraft of various weight classes. These models were used in simulations of flights to further develop TAP and the TASAR concept.

Unlike the kinematic aircraft performance models used for the Piaggio Avanti turboprop aircraft, the performance models for generic turbojet aircraft were primarily kinetic models that related thrust, drag, lift, and gravity in a total energy model. The implementation of two such different approaches to trajectory prediction within the same framework forced a careful examination of the design and enable the identification and elimination of assumptions that might limit the applicability of the software.

C. Compatibility with Performance Data from The Boeing Company

Through our relationship with Alaska Airlines in support of the operational trial of TAP described in Section V, we evaluated the possible use of aircraft performance data in the format used by the inflight performance software package of The Boeing Company. While some refinement of our design for performance function evaluation would be required, we determined that with the addition of a relatively small number of specialized C++ classes, Behavior-Based Trajectory Prediction is capable of using this type of data to predict trajectories.

At the time of this writing, The Boeing Company has not approved the use of data of this type for this purpose. Discussions concerning the possible use of such data in TAP are ongoing.

D. Adaptation of BADA Family 4 Aircraft Performance Data

As part of the overall investigation of ways to obtain accurate performance data for deployment on actual turbojet aircraft, we extended Behavior-Based Trajectory Prediction to accept data from the Base of Aircraft Data (BADA) Family 4.[5,6]

The performance data provided by BADA are intended for use in a total energy model substantially similar to that used by several other trajectory prediction algorithms, including the previously-implemented turbojet model of Behavior-Based Trajectory Prediction. Adapting to this data source was therefore easier than the implementation of the first turbojet models. We implemented new function evaluators suitable to evaluate mathematical functions of the forms defined by BADA Family 4, parsed the BADA aircraft performance model file to obtain the coefficients of the functions, and set these coefficients in the appropriate evaluators. We also eliminated table-based assumptions (largely in internal documentation and class-naming schemes) so that the new types of evaluators could be used on an equal basis with table-based evaluators.

The only large gap we found between the available performance data in BADA Family 4 and the requirements of Behavior-Based Trajectory Prediction for TAP was in regard to the estimation of ECON speeds. Although EUROCONTROL has published detailed algorithms that use a BADA aircraft performance model to calculate an optimal speed to meet a given cost index on a given trajectory,[7] what TAP requires in its trajectory prediction is not what the ECON speed *should* be under given conditions but rather what ECON speed *will* be selected by the aircraft's Flight Management System (FMS). With this in mind, the ideal model of ECON speed would be derived from the method used within the FMS itself; an alternative is to observe the ECON speeds chosen by the FMS under various conditions and construct an empirical model from those data. Using observations collected by Alaska Airlines with the relevant FMS, we constructed a suitable table of ECON speeds for a Boeing 737-900ER aircraft operating with the same model of FMS.

Two members of staff implemented the new features required to adapt to BADA Family 4 data (using a trivial preliminary model of ECON speed) in approximately one month. A smaller amount of labor sufficed to develop unit tests for the new functions, construct the table of observed ECON speeds, and test TAP in a simulation environment while using an actual BADA aircraft performance model.

We consider this experience to be an additional demonstration of the adaptability of Behavior-Based Trajectory Prediction to various data sources.

In discussions of the use of BADA Family 4 in TAP, EUROCONTROL has been very responsive on this matter and has supported our efforts by providing documentation and data. With the support of EUROCONTROL, both The Boeing Company and Airbus have approved the use of BADA Family 4 data for the operational trial of TAP described in Section V.

V. Future Plans

At the time of this writing, preparations are being made for an operational trial of TAP aboard selected Boeing 737-900ER aircraft operated by Alaska Airlines. This operational trial is expected to provide an opportunity to assess the quality of the predictions made by Behavior-Based Trajectory Prediction and to determine any refinements or corrections of the algorithm that may be needed.

More generally, since Behavior-Based Trajectory Prediction is designed to be extensible, we hope to explore additional variations of aircraft type, aircraft intent type, and requirements for trajectory output in the future.

VI. Conclusion

Behavior-Based Trajectory Prediction is an accurate, extensible, and highly adaptable method for predicting the trajectories of civil passenger aircraft, including the prediction of elapsed time and fuel consumed. Behavior-Based Trajectory Prediction fills a gap in the taxonomy of commonly-available trajectory prediction software between very accurate Flight Management Systems specialized to aircraft with specific guidance modes and more generic software that can predict trajectories for a variety of aircraft types but cannot accurately predict fuel consumption for an individual flight.

Acknowledgments

The authors thank Alaska Airlines, The Boeing Company, and EUROCONTROL for their cooperation and assistance in this work, without which a significant portion of its results would not have been possible.

References

[1] Ballin, M. G. and Wing, D. J., “Traffic Aware Strategic Aircrew Requests (TASAR),” *AIAA Aviation Technology, Integration, and Operations Conference*, AIAA-2012-5623, AIAA, Indianapolis, IN, Sept. 2012.

[2] Woods, S., Vivona, R. A., Henderson, J., Wing, D. J., and Burke, K. A., “Traffic Aware Planner for Cockpit-based Trajectory Optimization,” *AIAA Aviation and Aeronautics Forum and Exposition*, AIAA-2016-4067, AIAA, Washington, DC, June 2016.

[3] Karr, D. A., Vivona, R. A., Roscoe, D. A., DePascale, S. M., and Wing, D. J., “Autonomous Operations Planner: A Flexible Platform for Research in Flight-Deck Support for Airborne Self-Separation,” *AIAA Aviation Technology, Integration, and Operations Conference*, AIAA-2012-5417, AIAA, Indianapolis, IN, Sept. 2012.

[4] Vivona, R. A., Cate, K. T., and Green, S. M., “Abstraction Techniques for Capturing and Comparing Trajectory Predictor Capabilities and Requirements,” *AIAA Guidance, Navigation, and Control Conference*, AIAA-2008-7408, AIAA, Honolulu, HI, Aug. 2008.

[5] Nuic, A. and Mouillet, V., “User Manual for the Base of Aircraft Data (BADA) Family 4,” EEC Technical/Scientific Report 12/11/22-58, EUROCONTROL Experimental Centre, Brétigny-sur-Orge, Île-de-France, France, March 2016.

[6] Vivona, R. A., “NASA Traffic Aware Planner and BADA 4: Preparing for Commercial Flight Deck Operational Trials,” *BADA User Conference*, EUROCONTROL, Prestwick, Scotland, May 2017.

[7] Poles, D., Nuic, A., and Mouillet, V., “Advanced Aircraft Performance Modeling for ATM: Analysis of BADA Model Capabilities,” *IEEE/AIAA 29th Digital Avionics Systems Conference (DASC)*, IEEE, Salt Lake City, UT, Oct. 2010.