Model-based Prognostics of Hybrid Systems

Matthew Daigle[1], Indranil Roychoudhury[2], and Anibal Bregon[3]

[1] *NASA Ames Research Center, Moffett Field, California, 94035, USA*
matthew.j.daigle@nasa.gov

[2] *Stinger Ghaffarian Technologies Inc., NASA Ames Research Center, Moffett Field, California, 94035, USA*
indranil.roychoudhury@nasa.gov

[3] *Department of Computer Science, University of Valladolid, Valladolid, Spain*
anibal@infor.uva.es

ABSTRACT

Model-based prognostics has become a popular approach to solving the prognostics problem. However, almost all work has focused on prognostics of systems with continuous dynamics. In this paper, we extend the model-based prognostics framework to hybrid systems models that combine both continuous and discrete dynamics. In general, most systems are hybrid in nature, including those that combine physical processes with software. We generalize the model-based prognostics formulation to hybrid systems, and describe the challenges involved. We present a general approach for modeling hybrid systems, and overview methods for solving estimation and prediction in hybrid systems. As a case study, we consider the problem of conflict (i.e., loss of separation) prediction in the National Airspace System, in which the aircraft models are hybrid dynamical systems.

1. INTRODUCTION

Prognostics deals with predicting the occurrence of some system event, so that decisions can be made either autonomously or by human operators to improve system performance in some way. For example, in failure prognostics, end-of-life (EOL) is predicted so that system usage can be modified to extend system life or maintenance planned (Camci, 2009; Tian, Jin, Wu, & Ding, 2011; Saha & Goebel, 2009; Orchard & Vachtsevanos, 2009). Engineering systems are becoming increasingly complex, and often consist of a tight integration between hardware and software components. As such, most real-world engineering systems exhibit *hybrid* dynamics, i.e., a mix between continuous and discrete dynamics. This feature of modern-day systems makes prognostics more complex.

A significant amount of research has been performed dealing with hybrid systems in the areas of modeling, verification, diagnosis, and control, among others. Several modeling paradigms have been developed to represent hybrid system dynamics, such as hybrid automata (Henzinger, 2000) and hybrid bond graphs (P. J. Mosterman & Biswas, 1998; P. Mosterman & Biswas, 2000). A significant amount of research exists on diagnosis of hybrid systems (Narasimhan & Biswas, 2007; Narasimhan & Brownston, 2007; McIlraith, 2000; Koutsoukos et al., 2003; Hofbaur & Williams, 2004; Bayoudh et al., 2008; Bregon et al., 2011; Cocquempot et al., 2004; Daigle et al., 2010), yet little work exists on *prognosis* of hybrid systems. Only recently have approaches for hybrid systems prognostics been investigated (Chanthery & Ribot, 2013; Zabi et al., 2013; Gaudel et al., 2014; Yu et al., 2011). In (Chanthery & Ribot, 2013) and (Zabi et al., 2013), hybrid automata models are used, and in (Gaudel et al., 2014), a new formalism, hybrid particle petri nets, is introduced. These works are focused mainly on the integration of diagnosis and prognosis for hybrid systems. The prognosis aspect is limited in that it is focused specifically on the subproblem of failure prognostics, and, further, aging laws specifically take the form of Weibull models. In (Yu et al., 2011), hybrid bond graphs are used, but the approach is similarly limited.

In contrast, our aim is to develop a general, model-based prognostics framework for hybrid systems. We adopt a compositional, component-based modeling approach (Daigle, Bregon, & Roychoudhury, 2015). The modeling approach is inspired by hybrid bond graphs, but does not restrict component dynamics to a fixed set as with HBGs. We advance the theory of model-based prognostics to the more general formulation for hybrid systems, and describe the complexities introduced for prognostics with hybrid system models. Al-

gorithms for estimation and prediction using hybrid system models are also discussed. To demonstrate the approach, we look at the problem of conflict (i.e., loss of separation) prediction within the National Airspace System (NAS) (Erzberger, Paielli, Isaacson, & Eshow, 1997; Tomlin, Pappas, & Sastry, 1998).

The paper is organized as follows. Section 2 develops the model-based prognostics framework for hybrid systems. Section 3 discusses hybrid system estimation, and Section 4 covers the prediction problem. Section 5 develops the case study and demonstrates the approach. Section 6 concludes the paper.

2. MODEL-BASED PROGNOSTICS

In this section, we first describe our hybrid systems modeling paradigm. We then formulate the prognostics problem for hybrid systems, and present a computational architecture.

2.1. Hybrid Systems Modeling

We define hybrid system dynamics in a general compositional way, where the system is made up of a set of components. Each component is defined by a set of discrete modes, with a different set of constraints describing the continuous dynamics of the component in each mode. Here, system-level modes are defined implicitly through the composition of the component-level modes.

At the basic level, the continuous dynamics of a component in each mode are modeled using a set of *variables* and a set of *constraints*. A constraint is defined as follows:

Definition 1 (Constraint)**.** A constraint c is a tuple (ε_c, V_c), where ε_c is an equation involving variables V_c.

A component is defined by a set of constraints over a set of variables. The constraints are partitioned into different sets, one for each component mode. A component is then defined as follows:

Definition 2 (Component)**.** A *component* δ with n discrete modes is a tuple $\delta = (V_\delta, \mathcal{C}_\delta)$, where V_δ is a set of variables and $\mathcal{C}_\delta$ is a set of constraint sets involving variables in V_δ, where $\mathcal{C}_\delta$ is defined as $\mathcal{C}_\delta = \{C_\delta^1, C_\delta^2, \ldots, C_\delta^n\}$, with a constraint set, C_δ^m, defined for each mode $m = \{1, \ldots, n\}$.

By composing a set of components, we can define a system model as follows:

Definition 3 (Model)**.** A *model* $\mathcal{M} = \{\delta_1, \delta_2, \ldots, \delta_d\}$ is a finite set of d components for $d \in \mathbb{N}$.

Note that the set of variables for a model does not change with the mode, hence we need only a variable set in a component and not a set of variable sets as with constraints. The set of variables for a model, $V_\mathcal{M}$, is simply the union of all the component variable sets, i.e., for d components, $V_\mathcal{M} = V_{\delta_1} \cup V_{\delta_2} \cup \ldots \cup V_{\delta_d}$. We say that two components are connected if they share a variable, i.e., components δ_i and δ_j are connected if $V_{\delta_i} \cap V_{\delta_j} \neq \varnothing$. $V_\mathcal{M}$ consists of five disjoint sets, namely, the set of state variables, $X_\mathcal{M}$; the set of parameters, $\Theta_\mathcal{M}$; the set of inputs (variables not computed by any constraint), $U_\mathcal{M}$; the set of outputs (variables not used to compute any other variables), $Y_\mathcal{M}$; and the set of auxiliary variables, $A_\mathcal{M}$. Parameters, $\Theta_\mathcal{M}$, include explicit model parameters that are used in the model constraints (e.g., fault parameters). Auxiliary variables, $A_\mathcal{M}$, are additional variables that are algebraically related to the state, parameter, and input variables, and are used to simplify the structure of the equations.

The model constraints, $C_\mathcal{M}$, are a union of the component constraints over all modes, i.e., $C_\mathcal{M} = \mathcal{C}_{\delta_1} \cup \mathcal{C}_{\delta_2} \cup \ldots \cup \mathcal{C}_{\delta_d}$, where $\mathcal{C}_{\delta_i} = C_{\delta_i}^1 \cup C_{\delta_i}^2 \cup \ldots \cup C_{\delta_i}^n$ for n modes. Constraints are exclusive to components, that is, a constraint $c \in C_\mathcal{M}$ belongs to exactly one $\mathcal{C}_\delta$ for $\delta \in \mathcal{M}$.

To refer to a particular mode of a model we use the concept of a *mode vector*. A mode vector $\mathbf{m}$ specifies the current mode of each of the components of a model. So, the constraints for a mode $\mathbf{m}$ are denoted as $C_\mathcal{M}^\mathbf{m}$. For shorthand, we will refer to the modes only of the components with multiple modes.

The switching behavior of each component can be defined using a finite state machine or a similar type of control specification. The state transitions may be attributed to controlled or autonomous events. However, for the purposes of this paper, we view the switching behavior as a black box where the mode change event is given, and refer the reader to many of the approaches already proposed in the literature for modeling the switching behavior (Henzinger, 2000; P. J. Mosterman & Biswas, 1998). We distinguish between two types of mode changes, *controlled* and *autonomous*, where controlled mode changes depend solely on the system inputs, and autonomous mode changes depend also on internal system variables.

2.2. Problem Formulation

The system model is constructed as a collection of components describing the system variables and the constraints describing their relationships (Defn. 3). By collecting all the constraints over each mode, the model variables and equations can be presented in a summarized, abstract form, consisting of the (continuous) state vector, $\mathbf{x}(k) \in \mathbb{R}^{n_x}$; the mode (or, discrete state) vector, $\mathbf{m}(k) \in \mathbb{N}^{n_m}$; the unknown parameter vector, $\boldsymbol{\theta}(k) \in \mathbb{R}^{n_\theta}$; the input vector, $\mathbf{u}(k) \in \mathbb{R}^{n_u}$; the process noise vector, $\mathbf{v}(k) \in \mathbb{R}^{n_v}$; the output vector, $\mathbf{y}(k) \in \mathbb{R}^{n_y}$; the measurement noise vector, $\mathbf{n}(k) \in \mathbb{R}^{n_n}$; the state equation, $\mathbf{f}$; the mode transition equation, $\mathbf{g}$; and the

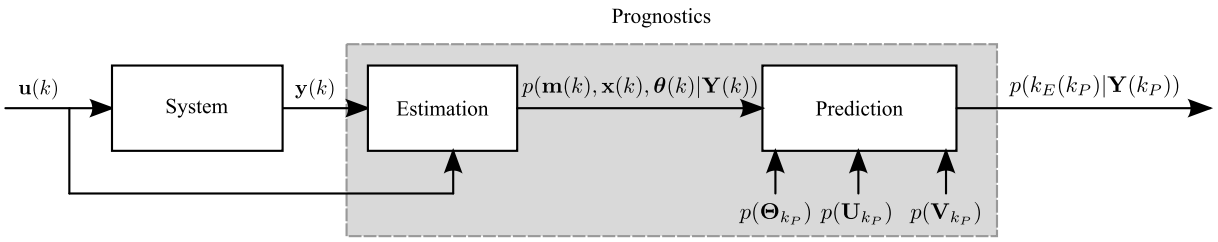

Figure 1. Model-based prognostics architecture.

output equation, $\mathbf{h}$:[1]

$$\mathbf{x}(k+1) = \mathbf{f}(k, \mathbf{m}(k), \mathbf{x}(k), \boldsymbol{\theta}(k), \mathbf{u}(k), \mathbf{v}(k)), \quad (1)$$

$$\mathbf{m}(k+1) = \mathbf{g}(k, \mathbf{m}(k), \mathbf{x}(k), \boldsymbol{\theta}(k), \mathbf{u}(k)), \quad (2)$$

$$\mathbf{y}(k) = \mathbf{h}(k, \mathbf{m}(k), \mathbf{x}(k), \boldsymbol{\theta}(k), \mathbf{u}(k), \mathbf{n}(k)), \quad (3)$$

where k is the discrete time variable. Here, the full state of the hybrid system is defined by both the continuous state $\mathbf{x}$ and the mode $\mathbf{m}$. Note that the unknown parameter vector $\boldsymbol{\theta}(k)$ is used to capture explicit model parameters whose values are unknown and time-varying stochastically. This presentation of the model is used here for problem formulation and does not necessarily represent implementation.

Prognostics is concerned with predicting the occurrence of some event E that is defined with respect to the mode, states, parameters, and inputs of the system. We define the event as the earliest instant that some event threshold $T_E : \mathbb{N}^{n_m} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_\theta} \times \mathbb{R}^{n_u} \to \mathbb{B}$, where $\mathbb{B} \triangleq \{0, 1\}$, changes from the value 0 to 1. That is, the time of the event k_E at some time of prediction k_P is defined as

$$k_E(k_P) \triangleq \inf\{k \in \mathbb{N} \colon k \geq k_P \wedge T_E(\mathbf{m}(k), \mathbf{x}(k), \boldsymbol{\theta}(k), \mathbf{u}(k)) = 1\}. \quad (4)$$

The time remaining until that event, Δk_E, is defined as

$$\Delta k_E(k_P) \triangleq k_E(k_P) - k_P. \quad (5)$$

2.3. Prognostics Architecture

We adopt a model-based prognostics architecture (Daigle & Goebel, 2013; Daigle & Sankararaman, 2013), in which there are two sequential problems, (*i*) the *estimation* problem, which requires determining a joint state-parameter estimate $p(\mathbf{m}(k), \mathbf{x}(k), \boldsymbol{\theta}(k)|\mathbf{Y}(k))$ based on the history of observations up to time k, $\mathbf{Y}(k) = [\mathbf{y}(k_0) \dots \mathbf{y}(k)]$, and (*ii*) the *prediction* problem, which determines at k_P, using the joint state-parameter estimate $p(\mathbf{m}(k_P), \mathbf{x}(k_P), \boldsymbol{\theta}(k_P)|\mathbf{Y}(k_P))$, the future parameter trajectory $p(\boldsymbol{\Theta}_{k_P})$, the future input trajectory $p(\mathbf{U}_{k_P})$, and the future process noise trajectory $p(\mathbf{V}_{k_P})$, a probability distribution $p(k_E(k_P)|\mathbf{Y}(k_P))$.

The prognostics architecture is shown in Fig. 1. In discrete time k, the system is provided with inputs $\mathbf{u}_k$ and provides measured outputs $\mathbf{y}_k$. The estimation module uses this information, along with the system model, to compute an estimate $p(\mathbf{m}(k), \mathbf{x}(k), \boldsymbol{\theta}(k)|\mathbf{Y}(k))$. The prediction module uses the joint state-parameter distribution and the system model, along with the distributions $p(\boldsymbol{\Theta}_{k_P})$, $p(\mathbf{U}_{k_P})$, and $p(\mathbf{V}_{k_P})$, to compute the probability distribution $p(k_E(k_P)|\mathbf{Y}(k_P))$. We describe the estimation problem in Section 3, and the prediction problem in Section 4.

3. ESTIMATION

As we have already mentioned, the *estimation* problem in our approach deals with determining a joint state-parameter estimate $p(\mathbf{m}(k), \mathbf{x}(k), \boldsymbol{\theta}(k)|\mathbf{Y}(k))$ based on the history of observations up to time k, $\mathbf{Y}(k)$. A general solution to the problem of joint state-parameter estimation is the unscented Kalman filter (UKF) (Julier & Uhlmann, 2004, 1997), which may be applied to nonlinear systems with Gaussian noise. Another solution highly used in the estimation literature is the particle filter (PF) (Arulampalam, Maskell, Gordon, & Clapp, 2002), which may be directly applied to nonlinear systems with non-Gaussian noise terms (Arulampalam et al., 2002). The main disadvantage of the PF against the UKF is the computational complexity, which is linear in the amount of samples, or *particles*, that are used to approximate the state distribution. The number of particles needed for joint state-parameter estimation is typically larger, and this number increases with the dimension of the state-parameter space.

For hybrid systems, the problem of joint state-parameter estimation becomes more complicated due to the mode changes. In the case the mode of the system is known, which happens when the system only has controlled mode changes, the problem reduces to continuous system state estimation, and the approaches mentioned above can be applied. A comparison of different filter techniques for the estimation problem in prognostics can be found in (Daigle, Saha, & Goebel, 2012). On the other hand, when autonomous mode changes can occur in the system, the estimation problem becomes more complex. Since the main focus of this work is the

[1] Bold typeface denotes vectors, and n_a denotes the length of a vector $\mathbf{a}$.

prediction problem of the prognostics architecture, we refer the reader to one of the many existing state estimation approaches in the literature (Rienmüller, Bayoudh, Hofbaur, & Travé-Massuyès, 2009; Benazera & Travé-Massuyès, 2009; Hofbaur & Williams, 2004; Koutsoukos et al., 2003; Blom & Bloem, 2004).

In (Hofbaur & Williams, 2004; Benazera & Travé-Massuyès, 2009) the authors propose an approach that requires to follow every possible state in the system at a particular instant, and then use search techniques from model-based reasoning in order to focus the estimation on the set of most likely modes, without missing symptoms that might be hidden among the system noise. Rienmuller et al. (Rienmüller et al., 2009) also use the state estimator proposed in (Hofbaur & Williams, 2004). In (Koutsoukos et al., 2003), the authors propose a particle filtering based estimation algorithm for a class of distributed hybrid systems. Finally, the approach in (Blom & Bloem, 2004) also uses particle filtering. Each one of this approaches can be perfectly integrated within our prognostics framework to precede the prediction problem.

4. PREDICTION

The prediction problem is to find the time of some system event E and/or system variables at the time of that event. In the model-based paradigm, the approach is conceptually straightforward. Given the system model, an initial state, and future inputs, we simulate the model forward in time until the event occurs. Of course, in practice, this is difficult because the inputs to this problem are all uncertain. In this context, the prediction problem becomes one of uncertainty propagation (Sankararaman, Daigle, Saxena, & Goebel, 2013; Sankararaman, Daigle, & Goebel, 2014).

At prediction time k_P, we consider four different inputs to the prediction problem: (*i*) the initial state estimate, $(\mathbf{m}(k_P), \mathbf{x}(k_P), \boldsymbol{\theta}(k_P))$; (*ii*) the future parameter trajectory, $\boldsymbol{\Theta}_{k_P}$ (where $\boldsymbol{\theta}(k_P)$ is the first value); (*iii*) the future input trajectory, $\mathbf{U}_{k_P}$; and (*iv*) the future process noise trajectory $\mathbf{V}_{k_P}$. The initial state estimate comes from the estimation module and is given as a probability distribution. The remaining uncertain inputs must be determined in some way either a priori or by using data from the system. For example, we can assume that the future inputs look like the past inputs over some time window with some uncertainty (Daigle, Saxena, & Goebel, 2012).

Underlying the overall prediction algorithm is the function P (Algorithm 1) that, given a realization of all the prediction inputs, computes the corresponding value of k_E. The algorithm simply simulates forward the model given the inputs. The version presented here extends the version for continuous systems (Daigle & Sankararaman, 2013) by including the discrete state, $\mathbf{m}(k)$, and the mode transition equation, $\mathbf{g}$ (Eq. 2). Further, it introduces a finite prediction horizon, specified by

Algorithm 1 $k_E(k_P) \leftarrow \mathtt{P}(\mathbf{m}(k_P),\ \mathbf{x}(k_P),\ \boldsymbol{\Theta}_{k_P},\ \mathbf{U}_{k_P},\ \mathbf{V}_{k_P}, k_H)$

$k \leftarrow k_P$
$\mathbf{x}(k) \leftarrow \mathbf{x}(k_P)$
while $T_E(\mathbf{m}(k), \mathbf{x}(k), \boldsymbol{\Theta}_{k_P}(k), \mathbf{U}_{k_P}(k)) = 0$ **or** $k < k_L$ **do**
  $\mathbf{x}(k+1) \leftarrow \mathbf{f}(k, \mathbf{m}(k), \mathbf{x}(k), \boldsymbol{\Theta}_{k_P}(k), \mathbf{U}_{k_P}(k), \mathbf{V}_{k_P}(k))$
  $\mathbf{m}(k+1) \leftarrow \mathbf{g}(k, \mathbf{m}(k), \mathbf{x}(k), \boldsymbol{\Theta}_{k_P}(k), \mathbf{U}_{k_P}(k))$
  $k \leftarrow k+1$
  $\mathbf{x}(k) \leftarrow \mathbf{x}(k+1)$
  $\mathbf{m}(k) \leftarrow \mathbf{m}(k+1)$
end while
if $k = k_H$ **then**
  $k_E(k_P) \leftarrow \infty$
else
  $k_E(k_P) \leftarrow k$
end if

k_H. If E is not reached by k_H, then ∞ is returned, meaning that E was not reached in the specified prediction horizon.

Prediction algorithms differ by how they make use of the P function. In Monte Carlo sampling, inputs are sampled stochastically from the given distributions, and P is called many times, once for each sample. Other algorithms like unscented transform sampling and the inverse first-order reliability method offer more complicated sampling or analytical schemes (Daigle & Sankararaman, 2013). For hybrid systems, these approaches become more complicated because there is a mix of discrete and continuous probability distributions. The presence of a discrete mode and the possibility of future mode changes means that the distribution for k_E will typically be multi-modal, which causes additional difficulties. The problem is present for both autonomous and controlled mode changes. Uncertainty in the initial state, process noise, and inputs means that, for some realizations, autonomous mode changes may occur, whereas for others it may not, and uncertainty in the future input trajectory means that controlled mode changes may occur for some realizations but not for others.

Although it can be inefficient due to the large number of sampling typically required, in this work we use Monte Carlo sampling, as it is the most straightforward to apply for hybrid systems; the algorithm can be directly used given distributions for all the prediction inputs. Since we consider a finite prediction horizon here, that limits the computational complexity of the algorithm. In the worst case, the complexity is given by the number of samples times the difference $k_H - k_P$.

5. CASE STUDY

The NAS consists of commercial flights, general aviation flights, air traffic controllers (ATCs), airports, etc. There are many prediction problems to consider within the NAS, in order to maintain safety and increase efficiency. We consider the problem of conflict prediction in the NAS. A conflict, or

loss of separation, is defined as an event in which two planes come within an unsafe distance from one another. In present-day operations, conflicts are predicted within a 20 minute prediction horizon (Erzberger et al., 1997). If a conflict is predicted, then an ATC must resolve it by issuing instructions to the aircraft to turn, change speed, or change altitude. We want to predict conflicts in order to determine if conflict-resolving maneuvers need to be issued. Such predictions serve as inputs to a decision-making problem. An estimate of uncertainty is also required in order to maintain safety and avoid risks.

5.1. Modeling

In order to apply model-based prognostics to this problem, we require a model of the system. In our modeling framework, each aircraft with its controller defines a component. The system is constructed as the set of all considered aircraft. We follow the approach in (Bilmoria, Banavar, Chatterji, Sheth, & Grabbe, 2000; Chatterji, Sridhar, & Bilimoria, 1996) for modeling the aircraft, using point-mass models with simplified dynamics. For each aircraft, we consider two components, one representing the aircraft dynamics, and one for the control. The dynamics always remain the same, but the control laws change depending on the flight phase (climb, cruise, or descent). The component equations are summarized in Table 1. While more complex and realistic equations may be used to describe the dynamics and control, the simplified equations here are actually quite close to what is used in practice, and suitable for demonstrating hybrid system prognosis concepts.

We assume a point-mass model as in most trajectory prediction methods in the literature (e.g., (Slattery & Zhao, 1997; Erzberger et al., 1997; Chatterji et al., 1996)). The aircraft position is defined by latitude λ, longitude τ, altitude h, heading χ (the angle on the horizontal plane), and flight path angle γ (the angle on the vertical plane). The velocity is described by the airspeed V_t and the climb velocity V_h. The groundspeed V_g is computed based on an addition of the airspeed and windspeed vectors. The windspeed vector is defined by magnitude V_w and an angle χ_w. We assume there is no vertical component to the wind. It can be decomposed into north and east components, W_N and W_E, respectively.

Latitude and longitude change based on the component of the airspeed and windspeed in the horizontal plane and the radius R, which is computed as the radius of the earth R_e and the altitude. Altitude changes directly based on the climb velocity. As in (Bilmoria et al., 2000), we assume simple dynamics where the airspeed, climb velocity, and heading change are based on abstracted first-order systems. For each, they change based on the error between the current value and a commanded value (c subscript) with some lumped inertia J.

The control laws aim to point the aircraft to the desired heading, and control the velocity according to the phase of flight. The desired heading angle is computed as the great-circle heading, χ_{GC} with a wind correction term $(\arcsin(V_w/V_t \sin(\chi_w - \chi_g)))$ (Chatterji et al., 1996). The desired climb velocity is zero in the cruise phase, and in the climb and descent phases it is determined with a proportional control law based on the altitude error. The commanded airspeed is fixed for climb and descent ($V_{t,climb}$ and $V_{t,descent}$, respectively), and for cruise it is determined based on the distance to be traveled, D, and the desired time to arrive at the waypoint, t^*. The desired waypoint is specified by desired latitude, longitude, altitude, and time of arrival, $(\lambda^*, \tau^*, h^*, t^*)$.

The mode transition equation is implemented as follows. In nominal operations, an aircraft goes from the climb mode, to cruise mode, then descent mode for the final waypoint (i.e., the arrival runway), which are autonomous mode transitions. The aircraft goes from climb or descent to cruise when it reaches the desired altitude h^* within some error bounds. It goes from cruise to climb or descent when the desired altitude is lower than the current altitude within some error bounds. Thus, in nominal operations, the aircraft follows the given waypoints and switches between the control modes based on these waypoints. If a conflict will occur, then an ATC action can also cause the aircraft to switch modes. Predictions will be made to determine if such actions need to take place.

The event of interest, E, is a conflict, which is defined as two aircraft being within 5 nautical miles and 1000 vertical feet of one another. Since this event is defined over multiple aircraft, we require a system-level perspective, where the system is the NAS or a specific region within it (Daigle, Bregon, & Roychoudhury, 2012).

The inputs to the system are the set of desired waypoints for each aircraft and the wind. We assume that there is no uncertainty in the desired waypoints. Without a model of how the wind changes, we assume that the windspeed and direction are steady within the 20 minute prediction horizon, and their values fall within some assumed probability distribution.

5.2. Demonstration of Approach

We consider a scenario in which to demonstrate the overall approach. Here, we have a region of the NAS in which five aircraft are heading to different navigation waypoints. For each aircraft, the probability of a conflict with another within the next 20 minutes is computed, as well as the estimated time of the conflict. Both the state and the wind are considered to be uncertain and captured through normal distributions. Process noise is ignored and parameters are assumed to be known. Monte Carlo sampling is used for prediction with 500 samples. Each sample is associated with a time of conflict, from which the probability distribution of conflict time is produced. The probability of a conflict is computed as the number of samples with finite conflict times (i.e., 20 min-

Table 1. Components of the NAS

Component	Mode	Constraints
Dynamics	1	$W_N{=}V_w \sin(\chi_w)$
		$W_E{=}V_w \cos(\chi_w)$
		$\dot{\lambda}_t{=}(V_t \cos\gamma \cos\chi + W_N)/R$
		$\dot{\tau}_t{=}(V_t \cos\gamma \sin\chi + W_E)/(R\cos\lambda)$
		$\dot{h}{=}V_h$
		$R{=}R_e + h$
		$\gamma{=}\arcsin(\dot{h}/V_t)$
		$\dot{V}_t{=}(V_{t,c} - V_t)/J_t$
		$\dot{V}_h{=}(V_{h,c} - V_h)/J_h$
		$\dot{\chi}{=}(\chi_c - \chi)/J_\chi$
		$V_{g,y}{=}V_t \cos\chi + V_w cos\chi_w$
		$V_{g,y}{=}V_t \cos\chi + V_w cos\chi_w$
		$V_g{=}V_t \cos\chi + V_w \cos\chi_w$
		$\chi_g{=}\arctan(V_{g,y}/V_{g,y})$
		$h{=}\int_{t_0}^{t} \dot{h}$
		$\lambda{=}\int_{t_0}^{t} \dot{\lambda}_t$
		$\tau{=}\int_{t_0}^{t} \dot{\tau}_t$
		$V_h{=}\int_{t_0}^{t} \dot{V}_h$
		$\chi{=}\int_{t_0}^{t} \dot{\chi}$
		$V_t{=}\int_{t_0}^{t} \dot{V}_t$
Control	1	$\chi_{GC}{=}\arctan \dfrac{\sin(\tau^* - \tau)\cos(\lambda^*)}{(\sin(\lambda^*)\cos(\lambda) - \sin(\lambda)\cos(\lambda^*)\cos(\tau^* - \tau))}$
		$V_{t,c}{=}V_{t,climb}$
		$V_{h,c}{=}(h^* - h)P_{h,climb}$
		$\chi_c{=}\chi_{GC} - \arcsin(V_w/V_t \sin(\chi_w - \chi_g))$
	2	$\chi_{GC}{=}\arctan \dfrac{\sin(\tau^* - \tau)\cos(\lambda^*)}{(\sin(\lambda^*)\cos(\lambda) - \sin(\lambda)\cos(\lambda^*)\cos(\tau^* - \tau))}$
		$D{=}\sqrt{(\lambda^* - \lambda)^2 R^2 + (\tau^* \cos(\lambda^*) - \tau\cos(\lambda))^2 R^2}$
		$V_{t,c}{=}D/(t^* - t)$
		$V_{h,c}{=}0$
		$\chi_c{=}\chi_{GC} - \arcsin(V_w/V_t \sin(\chi_w - \chi_g))$
	3	$\chi_{GC}{=}\arctan \dfrac{sin(\tau^* - \tau) * cos(\lambda^*)}{(sin(\lambda^*) * cos(\lambda) - sin(\lambda) * cos(\lambda^*) * cos(\tau^* - \tau))}$
		$V_{t,c}{=}V_{t,descent}$
		$V_{h,c}{=}(h^* - h)P_{h,descent}$
		$\chi_c{=}\chi_{GC} - \arcsin(V_w/V_t \sin(\chi_w - \chi_g))$

utes or less) over the total number of samples (which includes those for which no conflict occurred within 20 minutes, returned as ∞ by Algorithm 1).

The initial aircraft positions are shown in Fig. 2. Each aircraft is numbered, and their waypoints are drawn along with straight-line flight paths to their waypoints. The circle drawn around each aircraft represents the required separation distance (the aircraft are not drawn to scale). From the plot, it appears that conflicts may arise between aircraft 1 and aircraft 3, aircraft 2 and aircraft 3, and aircraft 4 and aircraft 5, as their intended flight paths cross. The predictions at $t = 0$ minutes are that aircraft 1 and aircraft 3 have a 76% probability of conflict, occurring between 13.3 and 15.7 minutes, that aircraft 2 and aircraft 3 have a 88% probability of conflict, occurring between 4.3 and 8.2 minutes, and that aircraft 4 and aircraft 5 have a 0.02% probability of conflict, occurring around 11 minutes.

Without intervention, the conflict between aircraft 2 and aircraft 3 appears at 6.6 minutes (see Fig. 3), so the original prediction captures the true conflict time. As the actual time of conflict is approached, the estimated probability of conflict increases, as shown in Fig. 4. After the actual conflict, the probability of conflict reduces. The drop to zero is not discontinuous, since there is uncertainty in the aircraft positions.

Without intervention, the conflict between aircraft 1 and aircraft 3 occurs at 14.6 minutes (see Fig. 5). The probability of conflict over time is shown in Fig. 6. Due to position uncertainty, the probability at the time of conflict is only around 50%, as the conflict is just on the border of not happening.

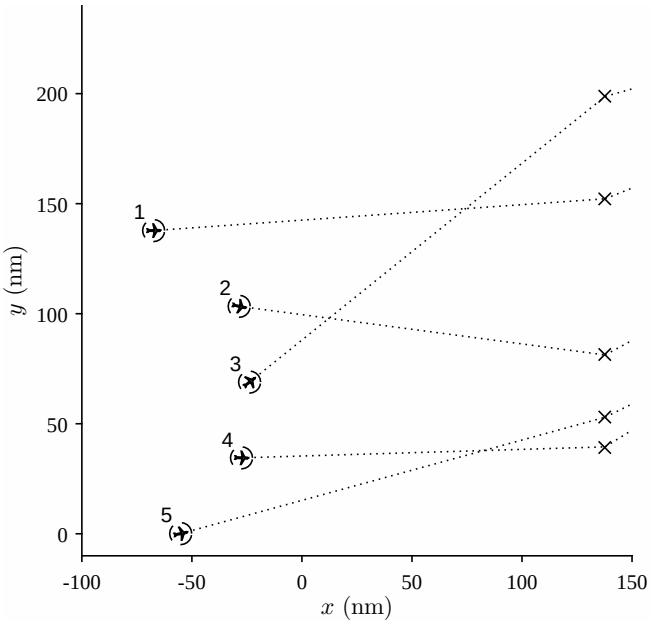

Figure 2. Aircraft positions at 0 minutes.

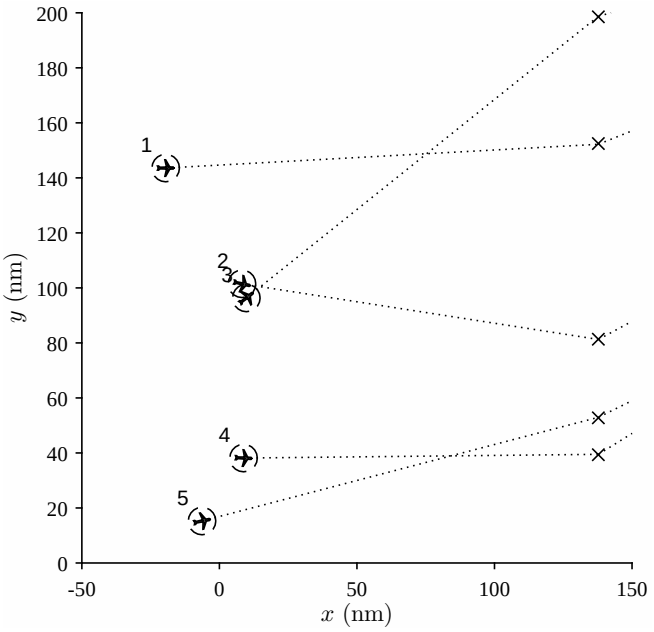

Figure 3. Aircraft positions at 6.6 minutes.

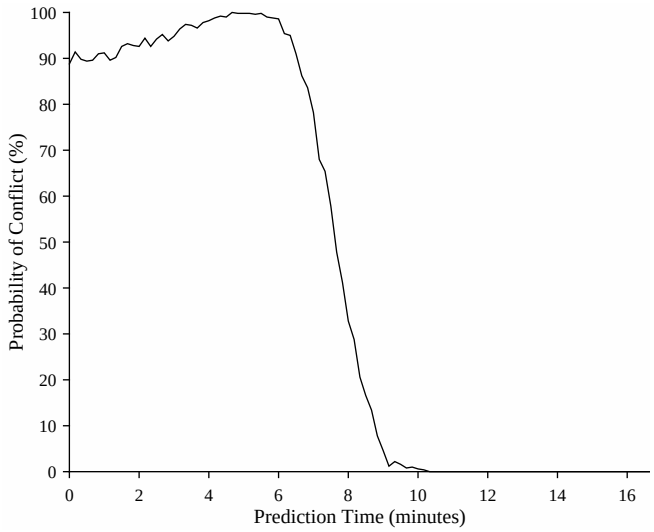

Figure 4. Predicted probability of conflict between 2 and 3 as a function of prediction time.

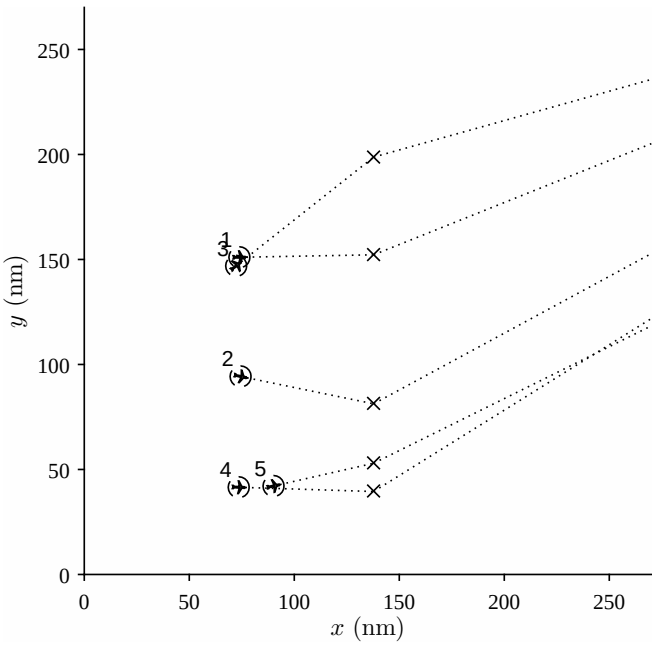

Figure 5. Aircraft positions at 14.6 minutes.

For aircraft 4 and aircraft 5, the probability of conflict is pretty low, and does not occur where the intended flight paths are (see Fig. 5, where aircraft 5 reaches the point where the paths cross before aircraft 4 does). The estimated probability of conflict rises to 100% around 12 minutes, but this is for a conflict that is to occur at $t = 30$ minutes where the intended flight paths to the next waypoints cross.

Here, the hybrid dynamics do not have much effect on the predictions, since all the conflicts happen to arise when in the cruise mode. If a conflict is to happen on the border of climb and cruise, for example, then the resulting distribution may become bimodal, for example if just on the border of the vertical separation being violated.

In actual operations, if a conflict is predicted within the 20 minute prediction horizon, a resolving maneuver would be issued by ATC in order to prevent the conflict. In current practice, only deterministic trajectory prediction is available, and errors are added on afterwards on top of these predic-

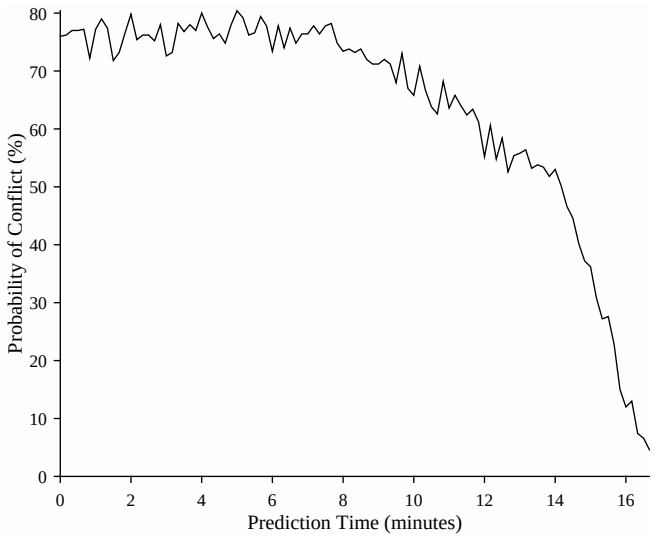

Figure 6. Predicted probability of conflict between 1 and 3 as a function of prediction time.

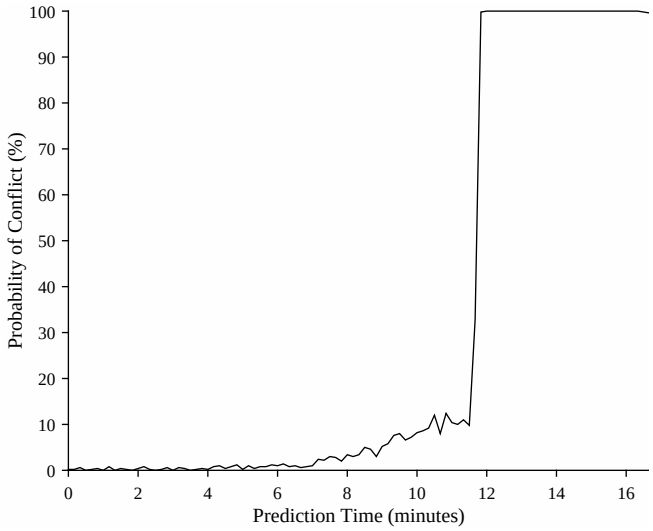

Figure 7. Predicted probability of conflict between 4 and 5 as a function of prediction time.

tions in an ad hoc manner to obtain probabilistic predictions of conflicts (Erzberger et al., 1997). On the other hand, in our approach, uncertainties are added at the time of prediction, and are propagated throughout the prediction procedure in order to obtain conflict probabilities. The model-based prognostics approach is a more rigorous method compared to the currently used ad hoc methods for treatment of uncertainty, and allows for more robust and efficient operations and more informed decision-making.

6. CONCLUSIONS

We presented an extension of the model-based prognosis framework to hybrid systems. The presence of mixed discrete and continuous dynamics presents significant challenges to the estimation and prediction problems. However, the overall model-based prognostics approach is essentially the same, as there is an estimation step followed by a prediction step. The only difference is the multi-modal, hybrid models used, which requires more complex estimation and prediction algorithms.

The approach was demonstrated on the problem of conflict prediction in the NAS. We showed how the framework can be applied, and discussed also how the model-based prognostics framework offers a more systematic and robust approach, especially in the context of handling uncertainty, than the current state-of-the-art in that domain. In fact, the framework presented here has a much broader applicability. For example, given models of unsafe weather systems, "conflicts" with unsafe weather regions can also be predicted, and these predictions can be used within decision-making algorithms. Other safety events, such as low fuel and high congestion, can also be predicted within this type of architecture.

While much work has been done on state estimation for hybrid systems, relatively little has been done on prediction and uncertainty propagation using these types of models. Monte Carlo sampling is a simple but inefficient solution to this problem, and future work will address more efficient prediction algorithms for hybrid systems, e.g., the extension of the unscented transform sampling method.

ACKNOWLEDGMENTS

This work has been funded by the NASA SMART-NAS project in the Airspace Operations and Safety Program of the Aeronautics Mission Directorate and the Spanish MINECO DPI2013-45414-R grant.

REFERENCES

Arulampalam, M. S., Maskell, S., Gordon, N., & Clapp, T. (2002). A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, *50*(2), 174–188.

Bayoudh, M., Trave-Massuyes, L., & Olive, X. (2008, July). Hybrid systems diagnosis by coupling continuous and discrete event techniques. In *Proceedings of the 17th International Federation of Automatic Control World Congress* (pp. 7265–7270).

Benazera, E., & Travé-Massuyès, L. (2009, October). Set-theoretic estimation of hybrid system configurations. *Trans. Sys. Man Cyber. Part B*, *39*, 1277–1291.

Bilmoria, K. D., Banavar, S., Chatterji, G. B., Sheth, K. S., & Grabbe, S. (2000, June). Facet: Future atm concepts

evaluation tool. In *3rd USA/EuropeATM R&D Seminar.*

Blom, H. A., & Bloem, E. A. (2004). Particle filtering for stochastic hybrid systems. In *43rd IEEE Conference on Decision and Control* (Vol. 3, pp. 3221–3226).

Bregon, A., Alonso, C., Biswas, G., Pulido, B., & Moya, N. (2011, October). Hybrid systems fault diagnosis with possible conflicts. In *Proceedings of the 22nd International Workshop on Principles of Diagnosis* (p. 195-202). Murnau, Germany.

Camci, F. (2009). System maintenance scheduling with prognostics information using genetic algorithm. *IEEE Transactions on Reliability*, *58*(3), 539–552.

Chanthery, E., & Ribot, P. (2013). An integrated framework for diagnosis and prognosis of hybrid systems. In *3rd Workshop on Hybrid Autonomous Systems* (pp. 14–25).

Chatterji, G., Sridhar, B., & Bilimoria, K. (1996, July). In *AIAA Guidance, Navigation, and Control and Conference.*

Cocquempot, V., El Mezyani, T., & Staroswiecki, M. (2004, July). Fault detection and isolation for hybrid systems using structured parity residuals. In *5th Asian Control Conference* (Vol. 2, p. 1204-1212). doi: 10.1109/ASCC.2004.185027

Daigle, M., Bregon, A., & Roychoudhury, I. (2012, September). A distributed approach to system-level prognostics. In *Annual Conference of the Prognostics and Health Management Society 2012* (p. 71-82).

Daigle, M., Bregon, A., & Roychoudhury, I. (2015, September). A structural model decomposition framework for hybrid systems diagnosis. In *26th International Workshop on Principles of Diagnosis.*

Daigle, M., & Goebel, K. (2013, May). Model-based prognostics with concurrent damage progression processes. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *43*(4), 535-546.

Daigle, M., Koutsoukos, X., & Biswas, G. (2010, October). An event-based approach to integrated parametric and discrete fault diagnosis in hybrid systems. *Trans. of the Institute of Measurement and Control*, *32*(5), 487-510.

Daigle, M., Saha, B., & Goebel, K. (2012, March). A comparison of filter-based approaches for model-based prognostics. In *Proceedings of the 2012 IEEE Aerospace Conference.*

Daigle, M., & Sankararaman, S. (2013, October). Advanced methods for determining prediction uncertainty in model-based prognostics with application to planetary rovers. In *Annual Conference of the Prognostics and Health Management Society 2013* (p. 262-274).

Daigle, M., Saxena, A., & Goebel, K. (2012, September). An efficient deterministic approach to model-based prediction uncertainty estimation. In *Annual Conference of the Prognostics and Health Management Society 2012* (p. 326-335).

Erzberger, H., Paielli, R. A., Isaacson, D. R., & Eshow, M. M. (1997). Conflict detection and resolution in the presence of prediction error. In *1st USA/Europe Air Traffic Management R&D Seminar.*

Gaudel, Q., Chanthery, E., & Ribot, P. (2014, September). Health monitoring of hybrid systems using hybrid particle petri nets. In *Annual Conference of the Prognostics and Health Management Society 2014.*

Henzinger, T. A. (2000). *The theory of hybrid automata*. Springer.

Hofbaur, M., & Williams, B. (2004, October). Hybrid estimation of complex systems. *IEEE Trans. on Systems, Man, and Cybernetics, Part B: Cybernetics*, *34*(5), 2178-2191.

Julier, S. J., & Uhlmann, J. K. (1997). A new extension of the Kalman filter to nonlinear systems. In *Proceedings of the 11th Intl. Symposium on Aerospace/Defense Sensing, Simulation and Controls* (pp. 182–193).

Julier, S. J., & Uhlmann, J. K. (2004, mar). Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, *92*(3), 401-422.

Koutsoukos, X., Kurien, J., & Zhao, F. (2003). Estimation of distributed hybrid systems using particle filtering methods. In *Hybrid Systems: Computation and Control (HSCC 2003). Springer Verlag Lecture Notes on Computer Science* (pp. 298–313). Springer.

McIlraith, S. (2000). Diagnosing hybrid systems: a Bayesian model selection approach. In *Proceedings of the 11th International Workshop on Principles of Diagnosis* (pp. 140–146).

Mosterman, P., & Biswas, G. (2000). A comprehensive methodology for building hybrid models of physical systems. *Artificial Intelligence*, *121*(1-2), 171 - 209.

Mosterman, P. J., & Biswas, G. (1998). A theory of discontinuities in physical system models. *Journal of the Franklin Institute*, *335*(3), 401–439.

Narasimhan, S., & Biswas, G. (2007, May). Model-Based Diagnosis of Hybrid Systems. *IEEE Trans. Syst. Man. Cy. Part A*, *37*(3), 348-361.

Narasimhan, S., & Brownston, L. (2007, May). HyDE: A general framework for stochastic and hybrid model-based diagnosis. In *Proc. of the 18th Int. WS. on Principles of Diagnosis* (p. 186-193).

Orchard, M., & Vachtsevanos, G. (2009, June). A particle filtering approach for on-line fault diagnosis and failure prognosis. *Transactions of the Institute of Measurement and Control*(3-4), 221-246.

Rienmüller, T., Bayoudh, M., Hofbaur, M., & Travé-Massuyès, L. (2009). Hybrid Estimation through Synergic Mode-Set Focusing. In *7th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes* (p. 1480-1485). Barcelona, Spain.

Saha, B., & Goebel, K. (2009, September). Modeling Li-ion

battery capacity depletion in a particle filtering framework. In *Proceedings of the Annual Conference of the Prognostics and Health Management Society 2009.*

Sankararaman, S., Daigle, M., & Goebel, K. (2014, June). Uncertainty quantification in remaining useful life prediction using first-order reliability methods. *IEEE Transactions on Reliability*, *63*(2), 603-619.

Sankararaman, S., Daigle, M., Saxena, A., & Goebel, K. (2013, March). Analytical algorithms to quantify the uncertainty in remaining useful life prediction. In *2013 IEEE Aerospace Conference.*

Slattery, R., & Zhao, Y. (1997). Trajectory synthesis for air traffic automation. *Journal of Guidance, Control, and Dynamics*, *20*(2), 232–238.

Tian, Z., Jin, T., Wu, B., & Ding, F. (2011). Condition based maintenance optimization for wind power generation systems under continuous monitoring. *Renewable Energy*, *36*(5), 1502–1509.

Tomlin, C., Pappas, G. J., & Sastry, S. (1998). Conflict resolution for air traffic management: A study in multiagent hybrid systems. *Automatic Control, IEEE Transactions on*, *43*(4), 509–521.

Yu, M., Wang, D., Luo, M., & Huang, L. (2011). Prognosis of hybrid systems with multiple incipient faults: augmented global analytical redundancy relations approach. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, *41*(3), 540–551.

Zabi, S., Ribot, P., & Chanthery, E. (2013, October). Health monitoring and prognosis of hybrid systems. In *Annual Conference of the Prognostics and Health Management Society.*

BIOGRAPHIES

Matthew Daigle received the B.S. degree in Computer Science and Computer and Systems Engineering from Rensselaer Polytechnic Institute, Troy, NY, in 2004, and the M.S. and Ph.D. degrees in Computer Science from Vanderbilt University, Nashville, TN, in 2006 and 2008, respectively. From September 2004 to May 2008, he was a Graduate Research Assistant with the Institute for Software Integrated Systems and Department of Electrical Engineering and Computer Science, Vanderbilt University, Nashville, TN. During the summers of 2006 and 2007, he was an intern with Mission Critical Technologies, Inc., at NASA Ames Research Center. From June 2008 to December 2011, he was an Associate Scientist with the University of California, Santa Cruz, at NASA Ames Research Center. Since January 2012, he has been with NASA Ames Research Center as a Research Computer Scientist. His current research interests include physics-based modeling, model-based diagnosis and prognosis, simulation, and hybrid systems. Dr. Daigle is a member of the Prognostics and Health Management Society and the IEEE.

Indranil Roychoudhury received the B.E. (Hons.) degree in Electrical and Electronics Engineering from Birla Institute of Technology and Science, Pilani, Rajasthan, India in 2004, and the M.S. and Ph.D. degrees in Computer Science from Vanderbilt University, Nashville, Tennessee, USA, in 2006 and 2009, respectively. Since August 2009, he has been with SGT, Inc., at NASA Ames Research Center as a Computer Scientist. His research interests include hybrid systems modeling, model-based diagnostics and prognostics, distributed diagnostics and prognostics, and Bayesian diagnostics of complex physical systems. Dr. Roychoudhury is a Senior Member of the IEEE and a member of the Prognostics and Health Management Society.

Anibal Bregon received his B.Sc., M.Sc., and Ph.D. degrees in Computer Science from the University of Valladolid, Spain, in 2005, 2007, and 2010, respectively. From September 2005 to June 2010, he was Graduate Research Assistant with the Intelligent Systems Group at the University of Valladolid, Spain. He has been visiting researcher at the Institute for Software Integrated Systems, Vanderbilt University, Nashville, TN, USA; the Dept. of Electrical Engineering, Linkoping University, Linkoping, Sweden; and the Diagnostics and Prognostics Group, NASA Ames Research Center, Mountain View, CA, USA. Since September 2010, he has been Assistant Professor and Research Scientist at the Department of Computer Science from the University of Valladolid. Dr. Bregon is a member of the Prognostics and Health Management Society and the IEEE. His current research interests include model-based reasoning for diagnosis, prognostics, health-management, and distributed diagnosis of complex physical systems.