An Integrated Model-Based Distributed Diagnosis and Prognosis Framework

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

[1] *University of Valladolid, Valladolid, Spain*
anibal@infor.uva.es
[2] *NASA Ames Research Center, Moffett Field, CA, 94035, USA*
matthew.j.daigle@nasa.gov
[3] *SGT Inc., NASA Ames Research Center, Moffett Field, CA, 94035, USA*
indranil.roychoudhury@nasa.gov

ABSTRACT

Diagnosis and prognosis are necessary tasks for system reconfiguration and fault-adaptive control in complex systems. Diagnosis consists of detection, isolation and identification of faults, while prognosis consists of prediction of the remaining useful life of systems. This paper presents an integrated model-based distributed diagnosis and prognosis framework, where system decomposition is used to perform the diagnosis and prognosis tasks in a distributed way. We show how different submodels can be automatically constructed to solve the local diagnosis and prognosis problems. We illustrate our approach using a simulated four-wheeled rover for different fault scenarios. Our experiments showed that our approach correctly performs fault diagnosis and prognosis in an efficient and robust manner.

1 INTRODUCTION

Systems health monitoring is essential to guaranteeing the safe, efficient, and reliable operation of engineering systems. Integrated systems health management methodologies include efficient fault diagnosis and prognosis mechanisms, where diagnosis involves *detecting* when a fault has occurred, *isolating* the true fault, and *identifying* the true damage to the system; and prognosis involves *predicting* how much useful life remains in the different components, subsystems, or systems given the diagnosed fault conditions. The information on the fault size and its expected impact on system life can be used to initiate recovery and reconfiguration actions that mitigate the fault or extend system life.

A large body of research exists for both diagnosis and prognosis methods. However, the integration of diagnosis and prognosis algorithms is seldom studied. In fact, many diagnosis methodologies leave out the fault identification step, which is necessary to perform a prediction from the current system state. Recently, we presented an integrated model-based framework for diagnosis and prognosis of complex systems, in which we made use of a common modeling framework for modeling both the nominal and faulty system behavior (Roychoudhury and Daigle, 2011).

In (Roychoudhury and Daigle, 2011), the nominal system behavior is estimated using an observer built with the nominal model. Faults are detected when a statistically significant deviation between the nominal estimates and the observed measurements is observed (Biswas *et al.*, 2003). Fault isolation compares the observed measurement deviations against predictions of how the measurements would deviate for each possible fault (Mosterman and Biswas, 1999). Fault identification performs joint state-parameter estimation using multiple observers, where, for each fault, the faulty system model is constructed as the nominal model integrated with a hypothesized fault model (Roychoudhury, 2009). The prognosis module uses for each fault hypothesis a prediction model based on its faulty system model and the identified fault parameters, to predict the remaining useful life of the system (Daigle and Goebel, 2011). However, this integrated solution performs the diagnosis and prognosis task in a centralized fashion, which is prone to single points of failures, and does not scale well as the size of the system increases.

To overcome such problems, in this work, we leverage previous results for distributed diagnosis (Bregon *et al.*, 2011) and distributed prognosis (**?**), which make use of structural model decomposition techniques, to provide a systematic approach to distributing the different diagnosis and prognosis steps presented in (Roychoudhury and Daigle, 2011). An algorithm, proposed in (Bregon *et al.*, 2011), is used to design local distributed subsystems based on global diagnosability analysis of the system, thus computing globally correct distributed diagnosis results without the use of a centralized coordinator. These local distributed subsystems are then used to construct local event-based distributed diagnosers for distributed fault isolation. Distributed fault identification is achieved by developing independent local state-parameter estimators for each hypothesized fault. Regarding distributed prediction, in (**?**) we developed an architecture that enables a large prognosis problem to be de-

composed into several independent local subproblems from which local results can be merged into a global result.

The main contribution of this paper is an integrated framework for distributed model-based diagnosis and prognosis. The proposed framework scales well and the resulting subproblems are typically small and easy to solve, resulting in an efficient and scalable distributed solution to the combined diagnosis and prognosis problem. We perform a number of experiments on a simulated four-wheeled rover testbed (Balaban *et al.*, 2011) to demonstrate and evaluate our approach.

The rest of the paper is organized as follows. Section 2 provides the problem formulation for our diagnosis and prognosis framework. Section 3 describes the distributed architecture and Section 4 briefly introduces its different components. Section 5 presents the case study and experimental results. Finally, Section 6 concludes the paper.

2 PROBLEM FORMULATION

The system model for nominal operation is represented as follows:

$$\dot{\mathbf{x}}(t) = \mathbf{f}(t, \mathbf{x}(t), \boldsymbol{\theta}(t), \mathbf{u}(t), \mathbf{v}(t)),$$
$$\mathbf{y}(t) = \mathbf{h}(t, \mathbf{x}(t), \boldsymbol{\theta}(t), \mathbf{u}(t), \mathbf{n}(t)),$$

where $\mathbf{x}(t) \in \mathbb{R}^{n_x}$ is the state vector, $\boldsymbol{\theta}(t) \in \mathbb{R}^{n_\theta}$ is the parameter vector, $\mathbf{u}(t) \in \mathbb{R}^{n_u}$ is the input vector, $\mathbf{v}(t) \in \mathbb{R}^{n_v}$ is the process noise vector, $\mathbf{f}$ is the state equation, $\mathbf{y}(t) \in \mathbb{R}^{n_y}$ is the output vector, $\mathbf{n}(t) \in \mathbb{R}^{n_n}$ is the measurement noise vector, and $\mathbf{h}$ is the output equation. The parameters $\boldsymbol{\theta}$ may evolve in arbitrary ways.[1]

Faults in the system are represented as changes in the above nominal system model. In this work, we only consider single faults occurring as changes in system parameters, $\boldsymbol{\theta}(t)$. We denote a fault, $f \in F$, as a tuple, (θ, g_f), where, $\theta \in \boldsymbol{\theta}$ is the faulty parameter, and g_f denotes the *fault progression function*, which models the way fault f is manifested in parameter θ, i.e.,

$$\dot{\theta}(t) = g_f(t, \mathbf{x}_f(t), \boldsymbol{\theta}_f(t), \mathbf{u}(t), \mathbf{m}_f(t)),$$

where $\mathbf{x}_f(t) = [\mathbf{x}(t), \ \theta(t)]^T$, $\boldsymbol{\theta}_f(t) = [\boldsymbol{\theta}(t) \backslash \{\theta(t)\}, \boldsymbol{\phi}_f(t)]^T$, $\boldsymbol{\phi}_f(t) \in \mathbb{R}^{n_{\phi_f}}$ is a vector of fault progression function parameters, and $\mathbf{m}_f(t) \in \mathbb{R}^{n_{m_f}}$ is a process noise vector associated with the fault progression function.

To develop our integrated diagnosis and prognosis framework, the faulty system model for fault $f = (\theta, g_f)$ involves integrating a single fault progression function into the nominal system model:

$$\dot{\mathbf{x}_f}(t) = \mathbf{f}_f(t, \mathbf{x}_f(t), \boldsymbol{\theta}_f(t), \mathbf{u}(t), \mathbf{v}(t)),$$
$$\mathbf{y}(t) = \mathbf{h}(t, \mathbf{x}(t), \boldsymbol{\theta}(t), \mathbf{u}(t), \mathbf{n}(t)),$$

where,

$$\mathbf{f}_f(\cdot) = \begin{bmatrix} \mathbf{f}(t, \mathbf{x}(t), \boldsymbol{\theta}(t), \mathbf{u}(t), \mathbf{v}(t)) \\ g_f(t, \mathbf{x}_f(t), \boldsymbol{\theta}_f(t), \mathbf{u}(t), \mathbf{m}(t)) \end{bmatrix} = \begin{bmatrix} \dot{\mathbf{x}}(t) \\ \dot{\theta}(t) \end{bmatrix}.$$

[1]Here, we use bold typeface to denote vectors, and use n_a to denote the length of a vector $\mathbf{a}$.

The goal of diagnosis is to: (*i*) detect a change in some $\theta \in \boldsymbol{\theta}$; (*ii*) isolate, under the single fault assumption, the true fault $f \in F$, i.e., both the parameter θ that has changed, and its fault progression function g_f; and (*iii*) identify (i.e. estimate) the fault damage by computing $p(\mathbf{x}_f(t), \boldsymbol{\theta}_f(t) | \mathbf{y}(0{:}t))$, where $\mathbf{y}(0{:}t)$ denotes all measurements observed up to time t.

The goal of prognosis is to determine the end of (useful) life (EOL) of a system, and/or its remaining useful life (RUL). For a given fault, f, using the fault estimate, $p(\mathbf{x}_f(t), \boldsymbol{\theta}_f(t) | \mathbf{y}(0{:}t))$, a probability distribution of EOL, $p(\text{EOL}_f(t_P) | \mathbf{y}(0{:}t_P))$, and/or RUL, $p(\text{RUL}_f(t_P) | \mathbf{y}(0{:}t_P))$ is computed at a given time point t_P (Daigle and Goebel, 2011). Since there is inherent uncertainty in the state-parameter estimate, process noise, and uncertainty in the future inputs, we predict a probability distribution rather than single EOL/RUL values. The acceptable behavior of the system is expressed through a set of n_c constraints, $C_{EOL} = \{c_i\}_{i=1}^{n_c}$, where $c_i : \mathbb{R}^{n_x} \times \mathbb{R}^{n_\theta} \times \mathbb{R}^{n_u} \rightarrow \mathbb{B}$ maps a given point in the joint state-parameter space given the current inputs, $(\mathbf{x}_f(t), \boldsymbol{\theta}_f(t), \mathbf{u}(t))$, to the Boolean domain $\mathbb{B} \triangleq [0, 1]$, where $c_i(\mathbf{x}_f(t), \boldsymbol{\theta}_f(t), \mathbf{u}(t)) = 1$ if the constraint is satisfied (Daigle and Goebel, 2011). If $c_i(\mathbf{x}_f(t), \boldsymbol{\theta}_f(t), \mathbf{u}(t)) = 0$, then the constraint is not satisfied, and the behavior of the system is deemed to be unacceptable. These individual constraints are combined into a single *threshold function* $T_{\text{EOL}_f} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_\theta} \times \mathbb{R}^{n_u} \rightarrow \mathbb{B}$, such that $T_{\text{EOL}_f}(\mathbf{x}_f(t), \boldsymbol{\theta}_f(t), \mathbf{u}(t)) = 1$ if these constraints are valid, and $T_{\text{EOL}_f}(\mathbf{x}_f(t), \boldsymbol{\theta}_f(t), \mathbf{u}(t)) = 0$ otherwise.

So, EOL_f may be defined as $\text{EOL}_f(t_P) \triangleq \inf\{t \in \mathbb{R} : t \geq t_P \text{ and } T_{\text{EOL}_f}(\mathbf{x}_f(t), \boldsymbol{\theta}_f(t), \mathbf{u}(t)) = 1\}$. i.e., EOL is the earliest time point at which the threshold is reached. RUL is expressed given EOL as $\text{RUL}_f(t_P) \triangleq \text{EOL}_f(t_P) - t_P$.

3 DISTRIBUTED ARCHITECTURE

For a large system, both the diagnosis and prognosis problems are correspondingly large. A centralized approach does not scale well, can be computationally expensive, and prone to single points of failures. Therefore, we propose to distribute the *global* integrated diagnosis and prognosis problem into independent *local* subproblems. In this work, we build on ideas from structural model decomposition (Blanke *et al.*, 2006; Pulido and Alonso-González, 2004) to compute local independent subproblems, which may be solved in parallel, thus providing scalability and efficiency.

Structural model decomposition allows decomposing a global model into a set of local models for which local diagnosis and prognosis problems can be directly defined. The global model of the system, denoted as $\mathcal{M}$, is defined as follows.

Definition 1 (Model)**.** The model of a system, $\mathcal{M}$, is a tuple $\mathcal{M} = (X, \Theta, U, Y, C)$, where X is the set of state variables of $\mathbf{x}$, Θ is the set of unknown parameters of $\boldsymbol{\theta}$, U is the set of input variables of $\mathbf{u}$, Y is the set of output variables of $\mathbf{y}$, and C is the set of model constraints of $\mathbf{f}$, $\mathbf{h}$, and C_{EOL}.

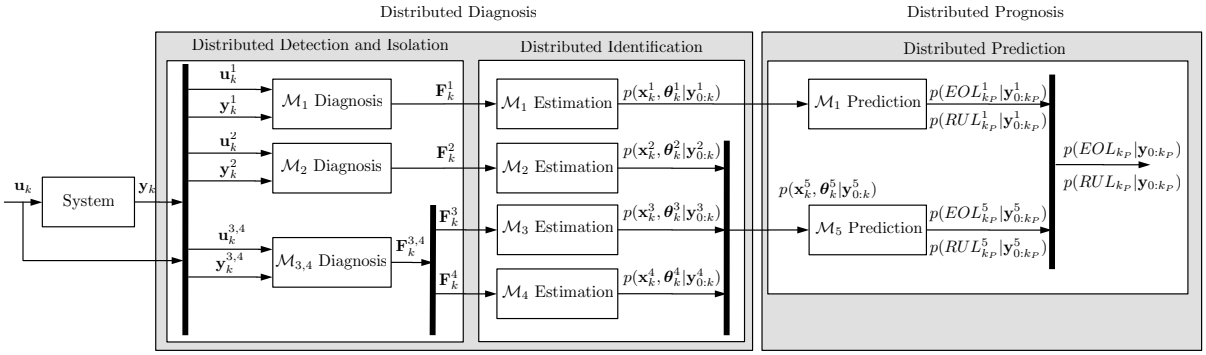

Figure 1: An instantiation of the integrated diagnosis and prognosis architecture.

Model decomposition is accomplished by using some variables (either measured variables or variables for which the values are known) as local inputs. Submodels computed in this way are computationally independent of each other. A submodel is then defined as follows:

Definition 2 (Submodel)**.** A submodel $\mathcal{M}_i$ of a system model $\mathcal{M} = (X, \Theta, U, Y, C)$ is a tuple $\mathcal{M}_i = (X_i, \Theta_i, U_i, Y_i, C_i)$, where $X_i \subseteq X$, $\Theta_i \subseteq \Theta$, $U_i \subseteq X \cup U \cup Y$, and $Y_i \subseteq Y$ are the state, parameter, input, and output variables, respectively, and $C_i \subseteq C$ are the submodel constraints.

In (Bregon *et al.*, 2011), we proposed a framework, based on structural model decomposition, to design submodels for distributed residual generation and fault isolation. Later, the framework was extended to cover identification in a distributed way. More recently, the same concept was used to decompose the prediction problem (Daigle *et al.*, 2011). Next, we discuss the fundamental ideas of our approach to obtain submodels for distributed diagnosis and prognosis. Then, we propose our integrated approach.

3.1 Model Decomposition for Distributed Diagnosis and Prognosis

For residual generation and fault isolation, the submodels use $U_i \subseteq U \cup (Y - Y_i)$, and we find a set of submodels such that each Y_i is a singleton, and over all Y_i, Y_j where $i \neq j$, $Y_i \cap Y_j = \varnothing$. So, each submodel uses some global model inputs and some measured values as local inputs so that the submodels become decoupled, are minimal, and may be computed independently from each other. An algorithm for computing the set of submodels with these properties is given in (Daigle *et al.*, 2011), which is based on the model decomposition algorithms presented in (Pulido and Alonso-González, 2004; Bregon *et al.*, 2012).

Based on the computed minimal submodel and an initial subsystem definition provided by the user, in (Bregon *et al.*, 2011) we propose a subsystem design approach which first computes the initial submodels for each subsystem, and then determines the minimal submodels that need to be merged with the initial submodels to create globally diagnosable subsystems.[2] Once the globally diagnosable subsystems have been designed, the merged submodels are used for distributed residual generation and to compute event-based local diagnosers for fault isolation. These design and diagnoser computation processes are detailed in (Bregon *et al.*, 2011). One of the properties of the design process is that the resulting local diagnosers are independent, and can provide globally correct diagnosis results without a centralized coordinator.

Regarding distributed fault identification, the joint state-parameter estimators are computed from the minimal submodels (Bregon *et al.*, 2012), i.e., we consider $U_i \subseteq U \cup (Y - Y_i)$ with Y_i as a singleton. It will be shown later that the fault identification module is the central part of our diagnosis-prognosis integration approach and provides the joint state-parameter estimations for the prediction module.

For distributed prediction, as detailed in (Daigle *et al.*, 2011), the submodels use $U_i \subseteq U_P$, where $U_P \subseteq X \cup U$. Here, U_P is a set of variables whose future values can be hypothesized, which depends on the hypothesized faults. We find a set of submodels such that each submodel has at least one $c \in C_{EOL}$ belonging to C_i, and over all submodels, all constraints in C_{EOL} are covered. This ensures that T_{EOL} may be computed for the system from the local constraints.

3.2 Distributed Architecture

Fig. 1 illustrates an example architecture for our distributed diagnosis and prognosis scheme. At each discrete time step, k, the system takes as input both $\mathbf{u}_k$ and $\mathbf{y}_k$ and splits them into local inputs $\mathbf{u}_k^i$ and local outputs $\mathbf{y}_k^i$ for the local diagnosers. Within each $\mathcal{M}_i$ local diagnoser, nominal tracking is performed, computing estimates of nominal measurements, $\hat{\mathbf{y}}_k^i$. The fault detector compares the estimated measurements against the observed measurements, to determine statistically significant deviations for the residual, $\mathbf{r}_k^i = \mathbf{y}_k^i - \hat{\mathbf{y}}_k^i$. Qualitative values of the deviations in the residuals are used by the event-based diagnoser to isolate faults. The set of isolated fault candidates $\mathbf{F}_k^i$ is used as input for the corresponding identification module. Identification is performed for each hypothesized fault, using the minimal submodels, to compute local state-parameter estimates $p(\mathbf{x}_k^i, \boldsymbol{\theta}_k^i | \mathbf{y}_{0:k}^i)$. These

[2] In this work, a subsystem is globally diagnosable if all faults in the subsystem are distinguishable from every other fault in the system using only local measurements.

local estimates are then used as input to the prediction submodels. In some cases, the local estimates have to be merged according to the prediction submodels. For example, submodel $\mathcal{M}_5$ uses the estimates from $\mathcal{M}_2$, $\mathcal{M}_3$, and $\mathcal{M}_4$, to build its local state. Distributed prediction modules compute local EOL/RUL predictions, $p(EOL^i_{k_P}|\mathbf{y}^i_{0:k_P})$ and $p(RUL^i_{k_P}|\mathbf{y}^i_{0:k_P})$ at given prediction time k_P based on the local EOL constraints. Finally, local predictions are combined into global predictions $p(EOL_{k_P}|\mathbf{y}_{0:k_P})$ and $p(RUL_{k_P}|\mathbf{y}_{0:k_P})$. The next section describes the details of the different modules of the distributed integrated diagnosis and prognosis architecture.

4 DIAGNOSIS AND PROGNOSIS APPROACH

Fig. 1 shows the basic modules of our distributed integrated approach. In this section we give details on how each module has been implemented, and establish the integration between the diagnosis and prognosis tasks.

4.1 Distributed Nominal Tracking

For distributed nominal tracking, each local diagnoser takes a subset of the local inputs $\mathbf{u}^i_k$ and local outputs $\mathbf{y}^i_k$, to compute an estimate of its output measurements $\hat{\mathbf{y}}^i_k$. Tracking is performed in discrete time using a robust filtering scheme, e.g., the extended or unscented Kalman filter (Julier and Uhlmann, 2004), the particle filter (Arulampalam *et al.*, 2002), among others, which provides accurate tracking in the presence of sensor noise, process noise, and discretization error.

4.2 Distributed Fault Detection

A statistical test is used to look for significant deviations in the residual signal $\mathbf{r}^i_k$. Residuals are computed as the difference between the estimated output measurements $\hat{\mathbf{y}}^i_k$ and the real measurements of the system $\mathbf{y}^i_k$. In our approach, we use a Z-test as described in (Biswas *et al.*, 2003).

4.3 Distributed Fault Isolation

Fault isolation is performed using local event-based diagnosers. These local event-based diagnosers are constructed from a set of globally diagnosable subsystems. The process to design globally diagnosable subsystems and construct the local event-based diagnosers is detailed in (Bregon *et al.*, 2011) and (Daigle *et al.*, 2009), respectively.

Fault isolation is triggered when a fault is detected, and it works as follows. Initially, all event-based local diagnosers start in their initial state, and the set of faulty candidates is empty. Local residual deviations cause the local diagnosers to move from one state to another. These residual deviations are abstracted to a tuple of qualitative symbols (σ_1, σ_2) for each residual signal, where σ_1 represents magnitude changes and σ_2 represents slope changes. A $+$ $(-)$ value indicates a change above (below) normal for a measurement residual or a positive (negative) residual slope. A 0 implies no change in the measurement value or a flat residual slope. The symbols are generated using a sliding window technique as described in detail in (Biswas *et al.*, 2003). If there is a match between an event from the current state and a tuple of qualitative symbols generated by any residual, the local diagnoser moves to the next state and remains active. If not, the local diagnoser blocks. This process continues until a local diagnoser reaches an accepting state. At this point, since the local diagnosers have been computed to be globally diagnosable, a globally correct diagnosis result has been found without the use of a centralized coordinator.

4.4 Distributed Fault Identification

In our distributed identification scheme, identification submodels, $\mathcal{M}_i$, are obtained, as explained in the previous section, as minimal submodels. A local state-parameter estimator is constructed for each identification submodel $\mathcal{M}_i$, and produces a local estimate $p(\mathbf{x}^i_k, \boldsymbol{\theta}^i_k|\mathbf{y}^i_{0:k})$ by using an appropriate algorithm. In this paper, we use an unscented Kalman filter (UKF) (Julier and Uhlmann, 2004) with a variance control algorithm (Daigle *et al.*, 2012). The UKF assumes the general nonlinear form of the state and output equations described in Section 2, but restricted to additive Gaussian noise.

4.5 Distributed Prediction

The local state-parameter estimates for each local distributed prediction module are constructed from the local estimates of the distributed fault identification submodels. Each prediction submodel is made up of a set of states X_i and parameters Θ_i, and constructs a local distribution $p(\mathbf{x}^i_k, \boldsymbol{\theta}^i_k|\mathbf{y}^i_{0:k})$, by assuming that the local state-parameter estimates are sufficiently represented by a mean $\boldsymbol{\mu}^i$ and covariance $\boldsymbol{\Sigma}^i$. For each prediction submodel $\mathcal{M}_i$, we combine the estimates of the local identification submodels that estimate states and parameters in $X_i \cup \Theta_i$ into $\boldsymbol{\mu}^i$ and $\boldsymbol{\Sigma}^i$. If two submodels estimate the same state variable or parameter, then many different techniques can be applied depending on the desired performance of the prediction submodels, e.g., taking the estimate with the smallest variance, or taking an average.

Different techniques can be applied to perform prediction for each prediction submodels. In this work, we use an unscented Kalman filter (Julier and Uhlmann, 2004). Given the mean and covariance information, we represent the distribution with a set of sigma points derived using the unscented transform. Then, each sigma point is simulated forward to EOL, and we recover the statistics of the EOL distribution given by the sigma points.

Algorithm 1 (Daigle and Goebel, 2011), shows the pseudocode for the prediction procedure. The algorithm is executed for each submodel i, deriving local EOL predictions using its local threshold function based on the local EOL constraints. For a given submodel, each sample j is propagated forward until $T^i_{EOL}(\mathbf{x}^{i(j)}_k, \boldsymbol{\theta}^{i(j)}_k)$ evaluates to 1. The algorithm hypothesizes future inputs $\hat{\mathbf{u}}^i_k$. Then, the global EOL/RUL is determined by the minimum of the local EOL/RUL distributions for each prediction submodel, i.e., $p(EOL^i_{k_P}|\mathbf{y}^i_{0:k_P})$ and $p(RUL^i_{k_P}|\mathbf{y}^i_{0:k_P})$. To compute this, we sample from each local EOL distribution

Algorithm 1 EOL Prediction

Inputs: $\{(\mathbf{x}_{k_P}^{i(j)}, \boldsymbol{\theta}_{k_P}^{i(j)}), w_{k_P}^{i(j)}\}_{j=1}^N$
Outputs: $\{EOL_{k_P}^{i(j)}, w_{k_P}^{i(j)}\}_{j=1}^N$
for $j = 1$ **to** N **do**
  $k \leftarrow k_P$
  $\mathbf{x}_k^{i(j)} \leftarrow \mathbf{x}_{k_P}^{i(j)}$
  $\boldsymbol{\theta}_k^{i(j)} \leftarrow \boldsymbol{\theta}_{k_P}^{i(j)}$
  while $T_{EOL}^i(\mathbf{x}_k^{i(j)}, \boldsymbol{\theta}_k^{i(j)}, \hat{\mathbf{u}}_k^i) = 0$ **do**
    Predict $\hat{\mathbf{u}}_k^i$
    $\boldsymbol{\theta}_{k+1}^{i(j)} \sim p(\boldsymbol{\theta}_{k+1}^i | \boldsymbol{\theta}_k^{i(j)})$
    $\mathbf{x}_{k+1}^{i(j)} \sim p(\mathbf{x}_{k+1}^i | \mathbf{x}_k^{i(j)}, \boldsymbol{\theta}_k^{i(j)}, \hat{\mathbf{u}}_k^i)$
    $k \leftarrow k + 1$
    $\mathbf{x}_k^{i(j)} \leftarrow \mathbf{x}_{k+1}^{i(j)}$
    $\boldsymbol{\theta}_k^{i(j)} \leftarrow \boldsymbol{\theta}_{k+1}^{i(j)}$
  $EOL_{k_P}^{i(j)} \leftarrow k$

and take the minimum of the local samples. This is repeated many times and the statistics of the global EOL distribution are computed.

5 CASE STUDY

In this section, we apply our system-level prognosis approach to a four-wheeled rover testbed developed at NASA Ames Research Center. We develop a model of the rover, and demonstrate the approach using simulated scenarios.

5.1 Rover Modeling

The rover model was originally presented in (Balaban *et al.*, 2011). In this section we summarize the main features and include some extensions to the model.

The rover consists of a symmetric rigid frame with four independently-driven wheels. The wheel speeds are governed by

$$\dot{\omega}_{FL} = \frac{1}{J_{FL}} \left(\tau_{mFL} - \tau_{fFL} - \tau_{glFL} + \tau_{grFL}\right) \quad (c_1)$$

$$\dot{\omega}_{FR} = \frac{1}{J_{FR}} \left(\tau_{mFR} - \tau_{fFR} - \tau_{glFR} - \tau_{grFR}\right) \quad (c_2)$$

$$\dot{\omega}_{BL} = \frac{1}{J_{BL}} \left(\tau_{mBL} - \tau_{fBL} - \tau_{glBL} + \tau_{grBL}\right) \quad (c_3)$$

$$\dot{\omega}_{BR} = \frac{1}{J_{BR}} \left(\tau_{mBR} - \tau_{fBR} - \tau_{glFR} - \tau_{grBR}\right). \quad (c_4)$$

The F, B, L, and R subscripts stand for *front*, *left*, *back*, and *right*, respectively. Here, for wheel w, J_w denotes the wheel inertia; τ_{mw} is the motor torque; $\tau_{fw} = \mu_{fw}\omega_w$ is the wheel friction torque, where μ_{fw} is a friction coefficient; $\tau_{glw} = r_w\mu_{gl}(v_w - v)$ is the torque due to slippage, where r_w is the wheel radius, μ_{gl} is a friction coefficient, v_w is the translational wheel velocity, and v is the translation velocity of the rover body; and $\tau_{grw} = r_w\mu_{grw}\omega\cos\gamma$ is the torque due to the rotational movement of the rover body, where μ_{grw} is a friction coefficient, ω is the rotational velocity of the rover body, and $\gamma = \arctan l/b$ with l being the rover's length and b being its width.

We consider here friction-based damage progression in the motors, resulting in an increase in motor friction over time. For wheel w, μ_{fw} is governed by (Daigle and Goebel, 2011)

$$\dot{\mu}_{fFL} = \nu_{fFL}\,\mu_{fFL}\,\omega_{FL}^2 \quad (c_5)$$

$$\dot{\mu}_{fFR} = \nu_{fFR}\,\mu_{fFR}\,\omega_{FR}^2 \quad (c_6)$$

$$\dot{\mu}_{fBL} = \nu_{fBL}\,\mu_{fBL}\,\omega_{BL}^2 \quad (c_7)$$

$$\dot{\mu}_{fBR} = \nu_{fBR}\,\mu_{fBR}\,\omega_{BR}^2, \quad (c_8)$$

where for wheel w, ν_{fw} is an unknown wear coefficient.

The translational velocity v of the rover is described by

$$\dot{v} = \frac{1}{m}\left(F_{glFL} + F_{glFR} + F_{glBL} + F_{glBR}\right), \quad (c_9)$$

where m is the rover mass, and for wheel w, $F_{glw} = \mu_{gl}(v_w - v)$ is the force due to slippage. The rotational velocity ω is described by

$$\dot{\omega} = \frac{1}{J}(d\cos\gamma F_{glFR} + d\cos\gamma F_{glBR} - d\cos\gamma F_{glFL} - d\cos\gamma F_{glBL} - dF_{grFL} - dF_{grFR} - dF_{grBL} - dF_{grBR}). \quad (c_{10})$$

Here, J is the rotational inertia of the rover and d is the distance from the center of the rover to each wheel.

The wheels are driven by DC motors with PID control that sets the voltages V applied to the motors. The motor currents i are governed by

$$\dot{i}_{FL} = \frac{1}{L}(V_{FL} - i_{FL}R_{FL} - k_\omega\omega_{FL}) \quad (c_{11})$$

$$\dot{i}_{FR} = \frac{1}{L}(V_{FR} - i_{FR}R_{FR} - k_\omega\omega_{FR}) \quad (c_{12})$$

$$\dot{i}_{BL} = \frac{1}{L}(V_{BL} - i_{BL}R_{BL} - k_\omega\omega_{BL}) \quad (c_{13})$$

$$\dot{i}_{BR} = \frac{1}{L}(V_{BR} - i_{BR}R_{BR} - k_\omega\omega_{BR}). \quad (c_{14})$$

Here, L is the motor inductance, R is the motor resistance, and k_ω is an energy transformation term. The motor torque is $\tau_{mw} = k_\tau i_w$, where k_τ is an energy transformation gain. The voltages applied to the motors are determined by the controllers, where for wheel w, $V_w = P * (u_w - \omega_w) + I * e_{iw}$, where P is a proportional gain, u_w is the commanded wheel speed, I is an integral gain, and e_{iw} is the integral error term. The integral error terms are governed by

$$\dot{e}_{iFL} = u_{FL} - \omega_{FL} \quad (c_{15})$$

$$\dot{e}_{iFR} = u_{FR} - \omega_{FR} \quad (c_{16})$$

$$\dot{e}_{iBL} = u_{BL} - \omega_{BL} \quad (c_{17})$$

$$\dot{e}_{iBR} = u_{BR} - \omega_{BR}. \quad (c_{18})$$

The batteries, which are connected in series, are described by a simple electrical circuit equivalent model that includes a large capacitance C_b in parallel with a resistance R_p, together in series with another resistance R_s.[3] The battery charge variables q_i are gov-

[3] We use a simple model here only for demonstration purposes. More detailed batter models for prognosis can be found in the literature, e.g., (Saha and Goebel, 2009).

erned by

$$\dot{q}_1 = -V_1/R_{p1} - (i_{FL} + i_{FR} + i_{BR} + i_{BL}) \quad (c_{19})$$
$$\dot{q}_2 = -V_2/R_{p2} - (i_{FL} + i_{FR} + i_{BR} + i_{BL}) \quad (c_{20})$$
$$\dot{q}_3 = -V_3/R_{p3} - (i_{FL} + i_{FR} + i_{BR} + i_{BL}) \quad (c_{21})$$
$$\dot{q}_4 = -V_4/R_{p4} - (i_{FL} + i_{FR} + i_{BR} + i_{BL}). \quad (c_{22})$$

The available sensors measure the voltages of the batteries,

$$V_1^* = q_1/C_{b1} - R_{s1} * (i_{FL} + i_{FR} + i_{BR} + i_{BL}) \quad (c_{23})$$
$$V_2^* = q_2/C_{b2} - R_{s2} * (i_{FL} + i_{FR} + i_{BR} + i_{BL}) \quad (c_{24})$$
$$V_3^* = q_3/C_{b3} - R_{s3} * (i_{FL} + i_{FR} + i_{BR} + i_{BL}) \quad (c_{25})$$
$$V_4^* = q_4/C_{b4} - R_{s4} * (i_{FL} + i_{FR} + i_{BR} + i_{BL}), \quad (c_{26})$$

the motor currents,

$$i_{FL}^* = i_{FL} \quad (c_{27})$$
$$i_{FR}^* = i_{FR} \quad (c_{28})$$
$$i_{BL}^* = i_{BL} \quad (c_{29})$$
$$i_{BR}^* = i_{BR}, \quad (c_{30})$$

and the wheel speeds,

$$\omega_{FL}^* = \omega_{FL} \quad (c_{31})$$
$$\omega_{FR}^* = \omega_{FR} \quad (c_{32})$$
$$\omega_{BL}^* = \omega_{BL} \quad (c_{33})$$
$$\omega_{BR}^* = \omega_{BR}. \quad (c_{34})$$

Here, the * superscript indicates a measured value.

We are interested in predicting when any of the rover batteries are at their voltage threshold, beyond which the batteries will be damaged (Saha and Goebel, 2009). The constraints are given as

$$V_1 > V^- \quad (c_{35})$$
$$V_2 > V^- \quad (c_{36})$$
$$V_3 > V^- \quad (c_{37})$$
$$V_4 > V^-, \quad (c_{38})$$

where the voltage threshold is given by $V^- = 9.6$ V, and for battery i, $V_i = q_i/C_{bi} - R_{si} * (i_{FL} + i_{FR} + i_{BR} + i_{BL})$. The rover cannot be operated when any of these constraints, c_{35}–c_{38}, are violated.

5.2 Results

To demonstrate the validity of the approach, we describe two different faulty scenarios of the rover. In the first, friction damage is progressing on one motor, and in the second, a capacitance decrease occurs in one battery. In all cases, the rover travels between various waypoints, moving at an average speed of 0.5 m/s. Table 1 shows the minimal submodels for the rover derived by using measured values as local inputs. Table 2 shows the submodels for residual generation and fault isolation. These submodels have been designed to obtain globally diagnosable subsystems by using the design algorithm in (Bregon *et al.*, 2011). For distributed fault identification we used the minimal submodels in Table 1. As will be shown, the correct prediction submodels to use depend on the scenario.

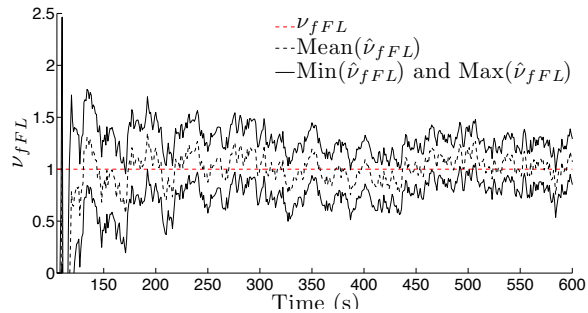

Figure 2: Estimated ν_{fFL} values.

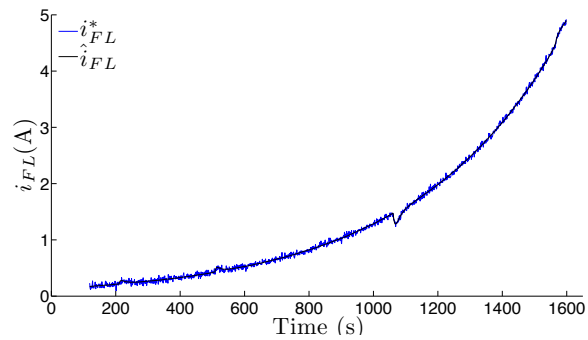

Figure 3: Current i_{FL} increase through time and estimated current increase.

Friction Damage Progression

We first consider a scenario in which, for the front-left motor, the friction begins to increase. The friction damage progression begins at $t = 50$ s with friction wear parameter $\nu_{fFL} = 1 \times 10^{-3}$. A fault is detected by the local diagnoser computed from submodel $\mathcal{M}_{5,9}$ at 119.25 s, via an increase in the motor current i_{FL}. The initial candidate list is immediately reduced to one candidate, $\{\nu_{fFL}\}$, based on the signatures and orderings. Thus the true fault is isolated.

Fault identification was initiated once the candidate was isolated. For the friction damage progression fault, the wear rate ν_{fFL} estimate averaged to $\nu_{fFL} = 1 \times 10^{-3}$ with very small output error. Figure 2 shows the wear parameter estimate for friction damage.

As a result of the continuously increasing friction, the current drawn by the motor increases as well in order for the motor controller to maintain the same desired wheel speed (Figure 3 shows this increase in the current through time). Hence, the total current drawn from the batteries is increased, and EOL occurs around half an hour. Because i_{FL} is constantly changing, and in a way that is dependent on the motor state, it is incorrect to use it as a local input for prediction and to decompose the prediction problem into independent local prediction problems for the batteries and motors, i.e., it is not known a priori. Therefore, we must use merged submodels, using as local inputs average values for the remaining motor currents, average commanded wheel speeds, and average rover translational velocity v and rotational velocity ω. The prediction

Submodel	X_i	Θ_i	U_i	Y_i	C_i
$\mathcal{M}_1$	q_1	C_{b1}	$i^*_{FL}, i^*_{FR}, i^*_{BL}, i^*_{BR}$	V^*_1	c_{19},c_{23},c_{27},c_{28},c_{29},c_{30}
$\mathcal{M}_2$	q_2	C_{b2}	$i^*_{FL}, i^*_{FR}, i^*_{BL}, i^*_{BR}$	V^*_2	c_{20},c_{24},c_{27},c_{28},c_{29},c_{30}
$\mathcal{M}_3$	q_3	C_{b3}	$i^*_{FL}, i^*_{FR}, i^*_{BL}, i^*_{BR}$	V^*_3	c_{21},c_{25},c_{27},c_{28},c_{29},c_{30}
$\mathcal{M}_4$	q_4	C_{b4}	$i^*_{FL}, i^*_{FR}, i^*_{BL}, i^*_{BR}$	V^*_4	c_{22},c_{26},c_{27},c_{28},c_{29},c_{30}
$\mathcal{M}_5$	i_{FL}, e_{iFL}	$\varnothing$	u_{FL}, ω^*_{FL}	i^*_{FL}	c_{11},c_{15},c_{27},c_{31}
$\mathcal{M}_6$	i_{FR}, e_{iFR}	$\varnothing$	u_{FR}, ω^*_{FR}	i^*_{FR}	c_{12},c_{16},c_{28},c_{32}
$\mathcal{M}_7$	i_{BL}, e_{iBL}	$\varnothing$	u_{BL}, ω^*_{BL}	i^*_{BL}	c_{13},c_{17},c_{29},c_{33}
$\mathcal{M}_8$	i_{BR}, e_{iBR}	$\varnothing$	u_{BR}, ω^*_{BR}	i^*_{BR}	c_{14},c_{18},c_{30},c_{34}
$\mathcal{M}_9$	$\omega_{FL}, v, \omega, \mu_{fFL}$	ν_{fFL}	$i^*_{FL}, \omega^*_{FR}, \omega^*_{BL}, \omega^*_{BR}$	ω^*_{FL}	c_1,c_5,c_9,c_{10},c_{27},c_{32},c_{33},c_{34}
$\mathcal{M}_{10}$	$\omega_{FR}, v, \omega, \mu_{fFR}$	ν_{fFR}	$i^*_{FR}, \omega^*_{FL}, \omega^*_{BL}, \omega^*_{BR}$	ω^*_{FR}	c_2,c_6,c_9,c_{10},c_{28},c_{31},c_{33},c_{34}
$\mathcal{M}_{11}$	$\omega_{BL}, v, \omega, \mu_{fBL}$	ν_{fBL}	$i^*_{BL}, \omega^*_{FL}, \omega^*_{FR}, \omega^*_{BR}$	ω^*_{BL}	c_3,c_7,c_9,c_{10},c_{29},c_{31},c_{32},c_{34}
$\mathcal{M}_{12}$	$\omega_{BR}, v, \omega, \mu_{fBR}$	ν_{fBR}	$i^*_{BR}, \omega^*_{FL}, \omega^*_{FR}, \omega^*_{BL}$	ω^*_{BR}	c_4,c_8,c_9,c_{10},c_{30},c_{31},c_{32},c_{33}

Table 1: Minimal and identification submodels.

Submodel	X_i	Θ_i	U_i	Y_i	C_i
$\mathcal{M}_{5,9}$	$\omega_{FL}, v, \omega, \mu_{fFL}, i_{FL}, e_{iFL}$	ν_{fFL}	$u_{FL}, \omega^*_{FR}, \omega^*_{BL}, \omega^*_{BR}$	ω^*_{FL}, i^*_{FL}	$C_5 \cup C_9$
$\mathcal{M}_{6,10}$	$\omega_{FR}, v, \omega, \mu_{fFR}, i_{FR}, e_{iFR}$	ν_{fFR}	$u_{FR}, \omega^*_{FL}, \omega^*_{BL}, \omega^*_{BR}$	ω^*_{FR}, i^*_{FR}	$C_6 \cup C_{10}$
$\mathcal{M}_{7,11}$	$\omega_{BL}, v, \omega, \mu_{fBL}, i_{BL}, e_{iBL}$	ν_{fBL}	$u_{BL}, \omega^*_{FL}, \omega^*_{FR}, \omega^*_{BR}$	ω^*_{BL}, i^*_{BL}	$C_7 \cup C_{11}$
$\mathcal{M}_{8,12}$	$\omega_{BR}, v, \omega, \mu_{fBR}, i_{BR}, e_{iBR}$	ν_{fBR}	$u_{BR}, \omega^*_{FL}, \omega^*_{FR}, \omega^*_{BL}$	ω^*_{BR}, i^*_{BR}	$C_8 \cup C_{12}$
$\mathcal{M}_{1,2,3,4}$	q_1, q_2, q_3, q_4	$C_{b1}, C_{b2}, C_{b3}, C_{b4}$	$i^*_{FL}, i^*_{FR}, i^*_{BL}, i^*_{BR}$	$V^*_1, V^*_2, V^*_3, V^*_4$	$C_1 \cup C_2 \cup C_3 \cup C_4$

Table 2: Residual generation and fault isolation submodels.

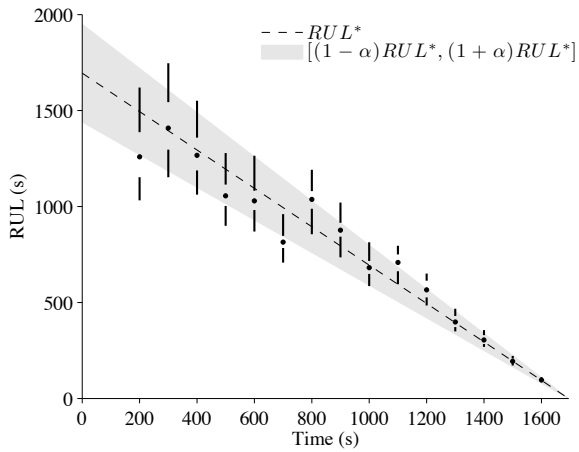

Figure 4: Predicted RUL of the rover. The mean is indicated with a dot and confidence intervals for 5% and 95% by lines. The gray cone depicts an accuracy requirement of 15%.

submodels for this case are shown in Table 3. Note that the prediction submodels used in this case do not correspond directly to those used for estimation. So, when constructing the estimate for $\mathcal{M}_{13}$, for example, it takes the estimates from $\mathcal{M}_1$ and $\mathcal{M}_9$.

The prediction results are shown in Fig. 4. The increased friction causes the batteries to discharge, and EOL occurs around 1650 s. Here, we used the relative accuracy (RA) as a measure of prediction accuracy, and the relative standard deviation (RSD) as a measure of spread as described in (Saxena *et al.*, 2010). For this experiment, RA averages to 91.63% and RSD averages to 16.26%.

Capacitance Decrease

As a second scenario, we consider a capacitance decrease fault in battery 3 of the rover, C_{b3}. The fault begins at $t = 50$ s with an abrupt decrease from 2000 to 1800 in the capacity of the battery. The fault is detected immediately by the local diagnoser computed from submodel $\mathcal{M}_{1,2,3,4}$ at 50.0 s, via an increase in the voltage V_3. The fault candidate is immediately isolated, $\{C_{b3}\}$, based on the signatures and orderings, thus starting the fault identification. For the capacitance fault, the estimated value of the capacitance averaged $C_{b3} = 1798.6$ with very small output error. As a result of the decrease in capacitance, the battery discharges at a faster rate, and so reaches end of discharge more quickly. Regarding prediction results for this experiment, RA averages to 85.42% and RSD averages to 12.98%.

6 CONCLUSIONS

This paper presented a distributed integrated model-based diagnosis and prognosis framework. Our approach starts off with a common modeling paradigm to model both the nominal behavior and fault progression, and then proposes a framework where the models are decomposed based on the requirements and constraints of each task. We demonstrated our approach on a four-wheeled rover testbed, where we diagnosed faults and prognosed the EOL/RUL accurately.

In future work, we will apply this approach to larger systems, to study the scalability of our diagnosis and prognosis scheme; and expand the capability of this approach to hybrid systems, as well as diagnosis and prognosis of multiple faults.

REFERENCES

(Arulampalam *et al.*, 2002) M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-Gaussian

Submodel	X_i	Θ_i	U_i	Y_i	C_i
$\mathcal{M}_{13}$	$q_1, i_{FL}, e_{iFL}, \omega_{FL}, \mu_{fFL}$	C_{b1}, ν_{fFL}	$u_{FL}, v, \omega, i_{FR}, i_{BL}, i_{BR}$	$\varnothing$	c_1,c_5,c_{11},c_{15},c_{19},c_{35}
$\mathcal{M}_{14}$	$q_2, i_{FL}, e_{iFL}, \omega_{FL}, \mu_{fFL}$	C_{b2}, ν_{fFL}	$u_{FL}, v, \omega, i_{FL}, i_{BL}, i_{BR}$	$\varnothing$	c_1,c_5,c_{11},c_{15},c_{20},c_{36}
$\mathcal{M}_{15}$	$q_3, i_{FL}, e_{iFL}, \omega_{FL}, \mu_{fFL}$	C_{b3}, ν_{fFL}	$u_{FL}, v, \omega, i_{FL}, i_{FR}, i_{BR}$	$\varnothing$	c_1,c_5,c_{11},c_{15},c_{21},c_{37}
$\mathcal{M}_{16}$	$q_4, i_{FL}, e_{iFL}, \omega_{FL}, \mu_{fFL}$	C_{b4}, ν_{fFL}	$u_{FL}, v, \omega, i_{FL}, i_{FR}, i_{BL}$	$\varnothing$	c_1,c_5,c_{11},c_{15},c_{22},c_{38}

Table 3: Prediction submodels using commanded wheel speeds and rover velocities as local inputs.

Submodel	X_i	Θ_i	U_i	Y_i	C_i
$\mathcal{M}_{17}$	q_1	C_{b1}	$i_{FL}, i_{FR}, i_{BL}, i_{BR}$	$\varnothing$	c_{19},c_{23},c_{35}
$\mathcal{M}_{18}$	q_2	C_{b2}	$i_{FL}, i_{FR}, i_{BL}, i_{BR}$	$\varnothing$	c_{20},c_{24},c_{36}
$\mathcal{M}_{19}$	q_3	C_{b3}	$i_{FL}, i_{FR}, i_{BL}, i_{BR}$	$\varnothing$	c_{21},c_{25},c_{37}
$\mathcal{M}_{20}$	q_4	C_{b4}	$i_{FL}, i_{FR}, i_{BL}, i_{BR}$	$\varnothing$	c_{22},c_{26},c_{38}

Table 4: Prediction submodels for capacitance faults.

Bayesian tracking. *IEEE Trans. on Signal Processing*, 50(2):174–188, 2002.

(Balaban *et al.*, 2011) E. Balaban, S. Narasimhan, M. Daigle, J. Celaya, I. Roychoudhury, B. Saha, S. Saha, and K. Goebel. A mobile robot testbed for prognostics-enabled autonomous decision making. In *Annual Conference of the Prognostics and Health Management Society*, pages 15–30, Montreal, Canada, September 2011.

(Biswas *et al.*, 2003) G. Biswas, G. Simon, N. Mahadevan, S. Narasimhan, J. Ramirez, and G. Karsai. A robust method for hybrid diagnosis of complex systems. In *Proceedings of the 5th Symposium on Fault Detection, Supervision and Safety for Technical Processes*, pages 1125–1131, June 2003.

(Blanke *et al.*, 2006) M. Blanke, M. Kinnaert, J. Lunze, and M. Staroswiecki. *Diagnosis and Fault-Tolerant Control*. Springer, 2006.

(Bregon *et al.*, 2011) A. Bregon, M. Daigle, I. Roychoudhury, G. Biswas, X. Koutsoukos, and B. Pulido. Improving Distributed Diagnosis Through Structural Model Decomposition. In *Proceedings of the 22nd International Workshop on Principles of Diagnosis*, pages 195–202, Murnau, Germany, Oct 2011.

(Bregon *et al.*, 2012) A. Bregon, G. Biswas, and B. Pulido. A Decomposition Method for Nonlinear Parameter Estimation in TRANSCEND. *IEEE Trans. Syst. Man. Cy. Part A*, 42(3):751–763, 2012.

(Daigle and Goebel, 2011) M. Daigle and K. Goebel. Multiple damage progression paths in model-based prognostics. In *Proc. of the 2011 IEEE Aerospace Conf.*, March 2011.

(Daigle *et al.*, 2009) M. J. Daigle, X. Koutsoukos, and G. Biswas. A qualitative event-based approach to continuous systems diagnosis. *IEEE Trans. on Control Systems Technology*, 17(4):780–793, July 2009.

(Daigle *et al.*, 2010) M. J. Daigle, I. Roychoudhury, G. Biswas, and X. Koutsoukos. A comprehensive diagnosis methodology for complex hybrid systems: A case study on spacecraft power distribution systems. *IEEE Trans. on System, Man, and Cybernetics, Part A*, 4(5):917 – 931, September 2010.

(Daigle *et al.*, 2011) M. Daigle, A. Bregon, and I. Roychoudhury. Distributed damage estimation for prognostics based on structural model decomposition. In *Proceedings of the Annual Conference of the Prognostics and Health Management Society 2011*, pages 198–208, September 2011.

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

(Julier and Uhlmann, 2004) S. J. Julier and J. K. Uhlmann. Unscented filtering and nonlinear estimation. *Proc. of the IEEE*, 92(3):401–422, March 2004.

(Mosterman and Biswas, 1999) P. J. Mosterman and G. Biswas. Diagnosis of continuous valued systems in transient operating regions. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Trans. on*, 29(6):554 – 565, November 1999.

(Pulido and Alonso-González, 2004) B. Pulido and C. Alonso-González. Possible conflicts: a compilation technique for consistency-based diagnosis. *IEEE Trans. on Systems, Man, and Cybernetics, Part B, Special Issue on Diagnosis of Complex Systems*, 34(5):2192–2206, 2004.

(Roychoudhury and Daigle, 2011) I. Roychoudhury and M. Daigle. An integrated model-based diagnostic and prognostic framework. In *Proceedings of the 22nd International Workshop on Principles of Diagnosis*, pages 44–51, October 2011.

(Roychoudhury, 2009) I. Roychoudhury. *Distributed Diagnosis of Continuous Systems: Global Diagnosis Through Local Analysis*. PhD thesis, Vanderbilt University, 2009.

(Saha and Goebel, 2009) B. Saha and K. Goebel. Modeling Li-ion battery capacity depletion in a particle filtering framework. In *Proc. of the Annual Conf. of the Prognostics and Health Management Society 2009*, September 2009.

(Saxena *et al.*, 2010) A. Saxena, J. Celaya, B. Saha, S. Saha, and K. Goebel. Metrics for offline evaluation of prognostic performance. *Int. Journal of Prognostics and Health Management*, 2010.