Development of a Mobile Robot Test Platform and Methods for Validation of Prognostics-Enabled Decision Making Algorithms

Edward Balaban[1], Sriram Narasimhan[2], Matthew J. Daigle[1], Indranil Roychoudhury[3], Adam Sweet[1], Christopher Bond[4], José R. Celaya[3], and George Gorospe[6]

[1] *NASA Ames Research Center, Moffett Field, CA, 94035, USA*
edward.balaban, matthew.j.daigle, adam.sweet@nasa.gov
[2] *University of California, Santa Cruz, NASA Ames Research Center, Moffett Field, CA, 94035, USA*
sriram.narasimhan@nasa.gov
[3] *SGT Inc., NASA Ames Research Center, Moffett Field, CA, 94035, USA*
indranil.roychoudhury@nasa.gov, jose.r.celaya@nasa.gov
[4] *NASA Kennedy Space Center, FL, 32899, USA*
christopher.n.bond@nasa.gov
[5] *USRA, NASA Ames Research Center, Moffett Field, CA, 94035*
george.e.gorospe@nasa.gov

ABSTRACT

As fault diagnosis and prognosis systems in aerospace applications become more capable, the ability to utilize information supplied by them becomes increasingly important. While certain types of vehicle health data can be effectively processed and acted upon by crew or support personnel, others, due to their complexity or time constraints, require either automated or semi-automated reasoning. Prognostics-enabled Decision Making (PDM) is an emerging research area that aims to integrate prognostic health information and knowledge about the future operating conditions into the process of selecting subsequent actions for the system. The newly developed PDM algorithms require suitable software and hardware platforms for testing under realistic fault scenarios. The paper describes the development of such a platform, based on the K11 planetary rover prototype. A variety of injectable fault modes are being investigated for electrical, mechanical, and power subsystems of the testbed, along with methods for data collection and processing. In addition to the hardware platform, a software simulator with matching capabilities has been developed. The simulator allows for prototyping and initial validation of the algorithms prior to their deployment on the K11. The simulator is also available to the PDM algorithms to assist with the reasoning process. A reference set of diagnostic, prognostic, and decision making algorithms is also described, followed by an overview of the current test scenarios and the results of their execution on the simulator.

1. INTRODUCTION

New designs of aerospace vehicles have been gradually gaining system health diagnostic and, in some cases, even prognostic capabilities (Reveley, Kurtoglu, Leone, Briggs, & Withrow, 2010), with the next logical step in autonomy maturation being decision-making based on such health information. A vehicle capable of evaluating its own health state and making (or assisting with making) decisions extending its remaining useful life or improving safety margins, may be able to go further and accomplish more objectives than a vehicle fully dependent on human actions. In addition to helping with mission execution, prognostics-enabled decision making systems may also prove valuable for vehicle maintenance, supply chain logistics, and fleet management.

While the research and development efforts for PDM systems are, for the most part, in their early stages, a need already exists for suitable platforms and techniqiues for their testing. At NASA Ames Research Center, test platforms from prior research efforts, such as (Poll et al., 2007) for electrical power systems or (Smith et al., 2009; Balaban et al., 2010) for electro-mechanical actuators, were created primarily with the diagnostic and prognostic elements of system health management in mind. In order to support our work in PDM a new platform was therefore needed. Such a platform is expected to support the following five high-level tasks: (*i*) development of system- and component-level PDM algorithms; (*ii*) development of realistic fault injection and accelerated aging techniques for algorithm testing; (*iii*) maturation and standardization of interfaces between reasoning algorithms; (*iv*) performance comparison of PDM algorithms from different

sources; and (v) generation of publicly available datasets for enabling further PDM research.

While an aerial test vehicle, such as an unmanned fixed-wing airplane or a helicopter, would allow testing of PDM technologies on complex scenarios and with motion in three-dimensional space, operating it is often expensive. A single flight hour often requires many days of preparation. Safety requirements for an aerial vehicle, whether manned or unmanned, are also usually quite stringent. In contrast, a ground vehicle can operate at low speeds in a controlled environment, making experiments easier, faster, and safer to set up. The experiments can still involve motion, complex subsystems interactions, and elaborate mission plans, however the possibility of a dangerous situation occurring is reduced significantly. For technologies in their early phases of development in particular, a ground vehicle platform could provide a suitable test environment for the majority of development goals at a fraction of the cost of an aerial vehicle (and usually with a clear transition path to the latter).

This paper updates and expands our previous description of the effort to develop such a platform and the associated PDM validation techniques (Balaban, Narasimhan, et al., 2011). The new features of the K11 testbed (a planetary rover prototype) include a redesigned sensor suite and a more capable battery management system. The K11 software simulator has also been updated with more accurate physics models. A new software architecture has been deployed on the testbed, supporting rapid integration of reasoning algorithms. The paper also provides an updated description of the reasoning algorithms currently under development and presents the formulation and results from the latest set of validation scenarios.

This paper is organized as follows. Section 2, describes related efforts. Section 3 presents the hardware and system software of the K11, while Section 4 describes the simulator and the experimental validation of the physics models. Section 7 presents the reasoning algorithms (diagnostic, prognostic, and decision making) currently being developed on the K11. The fault management experiments executed on the K11 simulator and their results are the subject of Section 8. Finally, Section 9 provides a summary of the accomplishments and outlines directions for future work.

2. RELATED WORK

Brown, Georgoulas, and Bole (2009) report on a prognostics-enhanced fault-tolerant controller that trades off performance for remaining useful life. The controller is based on model-predictive control principles, with control boundaries for a given remaining useful life estimate (corresponding to a particular input) used as soft cost constraints. The work is extended with error analysis and estimation of uncertainty bounds for long-term Remaining Useful Life (RUL) predictions in (Brown & Vachtsevanos, 2011). In (Bole, Tang, Goebel, & Vachtsevanos, 2011) the authors also study the optimal load allocation problem given prognostic data on fault magnitude growth. *Value at Risk*, coming from the field of finance, is used as the key performance metric. The case study used for the experiments is an unmanned ground vehicle (UGV), where winding insulation on the drive motors is degrading due to thermal stress.

The work done by Tang, Edwards, Orchard, and others on Automated Contingency Management (ACM) includes elements of prognostics-enhanced control, but also extends to mission replanning based on prognostic information (Tang et al., 2007; Edwards, Orchard, Tang, Goebel, & Vachtsevanos, 2010; Tang, Hettler, Zhang, & Decastro, 2011). Diagnostic and prognostic algorithms for various types of components are developed and integrated into a prototype UGV decision-making framework. RUL estimates are used either as a constraint or an additional element in the cost function in the UGV path planning algorithm. A *Field D**-style search algorithm is used for receding horizon planning. Methods for estimating and managing uncertainty are also developed.

3. TESTBED

The K11 (shown in Figure 1) is a four-wheeled rover, originally designed as a platform for testing power-efficient rover designs in Antarctic conditions (Lachat, Krebs, Thueer, & Siegwart, 2006). Since being repurposed for PDM research, the rover has been substantially updated, with its power distribution system, sensor suite, data acquisition module, and system software redesigned from the ground up. The following subsections provide more details on the K11 hardware, software, and fault injection methods (both those methods that have already been implemented and those that are planned to be added in the near future).

3.1. Hardware

The rover is roughly 1.4 m long by 1.1 m wide by 0.63 m tall. Its chassis is a lightweight H-structure with a joint around the front roll axis to ensure that the wheels stay in contact with the ground on uneven terrain. Each wheel is driven by an independent 250 W graphite-brush motor, connected through a bearing and gearhead system, with control performed by a single-axis digital motion controller. The rover is powered by twenty four 2.2 Ah lithium-ion single cell batteries (18650 form factor), organized in two parallel strings of twelve batteries in series. An onboard laptop computer runs the control and data acquisition software, as well as the reasoning algorithms. A second laptop computer currently serves as a ground control station.

The battery management system (BMS) provides battery charging and load balancing capabilities. The BMS also sends voltage and temperature measurements for each of the individual cells to the onboard computer via a Controller Area

Network (CAN) bus interface. The data acquisition (DAQ) module collects current and motor temperature measurements and sends them to the onboard computer. The motor controllers send back motion data such as encoder position values, commanded speed, and actual speed. Finally, a Google Nexus S smartphone is utilized as an additional sensor suite. It contains a GPS receiver, a gyroscope, an accelerometer, a photo and video camera, a magnetic compass, as well as data processing and storage resources. The phone also has a built-in wireless capability for communicating with other on-board components (and directly with the ground station). Table 1 presents the measurements available on the K11.

Figure 1. The K11 rover

3.2. Software

The K11 testbed software provides navigation, inter-module communication (middleware), and telemetry functions. The smartphone hosts a data acquisition module that collects data from the phone's sensors and sends it over a User Datagram Protocol (UDP) socket to the onboard computer. The central data acquisition software, running on the computer, receives the data, merges it with the data received from the DAQ system and the motor controllers, and records it in a unified data file. A unified data stream is also transmitted to the ground control station. The central data acquisition software on the K11 is based on LabView from National Instruments. The user interface on the ground station is also created in LabView, and, among other functions, allows the operator to take manual control.

Integration between the rover testbed (or its simulator) and the reasoning algorithms is accomplished through a publish/subscribe architecture. The architecture is implemented through the Internet Communication Engine, ICE (Henning, 2004). Standardized interface definition files are used to describe messages exchanged among the software and hardware modules. The message types include command inputs, sensor data, vehicle state information, fault diagnosis candidates, as well as unordered and ordered waypoint lists. A central server coordinates message exchanges among any number of devices on the same network. In order to be integrated into the architecture, a new reasoning module needs to only implement a minimal interface code necessary to register with the ICE server and to publish/subscribe to the appropriate messages. For example, a diagnostic module would subscribe to rover commands and sensor data and publish diagnostic messages. Thus the architecture allows for easy accommodation of modules implemented in different programming languages and running on dissimilar platforms.

Table 1. Measurements available on the K11

Measurement Type	Comments	Units
GPS	Longitude and latitude	deg
Gyroscope	Roll, pitch, yaw rates	rad/sec
Accelerometer	3-axis acceleration	m/sec^2
Magnetometer	3-axis magnetic field	μT
Motor temperature	On each motor (to be implemented)	deg C
Battery temperature	On each individual battery cell	deg C
Battery voltage	On each individual battery cell	V
Position encoder	On each individual motor	counts
Total current	A current sensor on the power bus from the battery to the motor controllers	A
Individual motor current	Custom sensors on the lines from the power bus to the motors	A

3.3. Fault Injection and Accelerated Aging Techniques

A number of fault modes have been selected for implementation on the K11 testbed, summarized in Table 2. The criteria for their selection include relevance to a variety of aerospace vehicle types, feasibility of implementation, and the progression time from fault to failure. If the progression time is too brief, then a meaningful prediction of remaining useful life (and, consequently, actions based on it) may not be possible. On the other hand, if the fault-to-failure progression time is measured in years, then executing experiments for those fault modes is impractical. While faults in these two categories are outside of the scope of this research, that certainly does not mean that they cannot (or should not) be handled by a health management system. Abrupt failures, unless resulting in a complete loss of control over the vehicle, can still be detected and identified by a diagnoser, then analyzed and acted upon by a decision-making system. Long duration

fault modes (e.g. airframe component fatigue) can be modeled with detailed simulations of future loads and operating conditions. Ideally, long-term degradation would be tracked from the time the vehicle (or a specific component) is new, in order to allow for a better estimation of the current health state.

Several modes described in Table 2 were selected for the experiments in the initial phase of the project. The methods for modeling progression of these faults are described in Section 4. The techniques (current and planned) for injecting the faults on the hardware testbed are described next.

Table 2. Potential fault types

Fault Mode	Injection Method	Subsystem
Battery capacity degradation	Accelerated aging	Power
Battery charge depletion	Discharging	Power
Parasitic electric load	Programmable	Power distribution
Motor driver faults	Electronic components replacement with aged units	Power distribution
Motor controller failure	Software	Electro-mechanical
Increased motor friction	Mechanical brake	Electro-mechanical
Sensor (bias, drift, scaling, or failure)	Software	Sensors

Out of the modes listed in Table 2, battery charge depletion is expected to have the shortest time of progression to failure (a few hours of continuous use at the most). The faults with the longest time-to-failure values are expected to be the electronic component faults, possibly taking on the order of months to fail (assuming the current fault seeding techniques). The other fault modes fall somewhere in between these two cases.

3.3.1. Remaining Battery Charge Tracking

While not being, in the strict sense, a fault, tracking the remaining battery charge will be one of the main tasks for the prognostic system. End-of-charge is an end-of-life criterion, so the remaining charge estimate will be an important factor for the PDM software. Most battery-powered vehicles incorporate some form of state-of-charge (SOC) monitoring. Usually SOC monitoring is based on Coulomb counting, i.e. integrating the current drawn over time, divided by the rated capacity of the battery. We will use a somewhat more robust, model-based approach, described in Section 7.2 (see also (Saha & Goebel, 2009; Saha, Quach, & Goebel, 2012)).

3.3.2. Battery Capacity Degradation

As the rover batteries go through charge/discharge cycles, their capacity to hold charge diminishes. The degradation rate will depend on several factors, such as imposed loads, environmental conditions, and charge procedures. For example, lithium-ion chemistry batteries undergo higher rates of capacity fade with higher current draw and operational temperatures. Even at rest, this type of battery has chemical processes occurring that have long-term effects - for instance, latent self-discharge and transient recovery during relaxation (Huggins, 2008). The depth-of-discharge and even the storage temperature have major influences on the overall life of the battery as well. The batteries will age naturally in the course of rover operations, and one method of injecting this fault is to use older, degraded batteries on the rover. The battery aging test stand (Saha & Goebel, 2009) at NASA Ames will be used to age rover battery cells to a desired point in their life cycle.

3.3.3. Parasitic Load

A parasitic electrical load will be injected on the main power distribution line via a remotely controlled rheostat. The rheostat can be set for resistance from 0 to 100 Ohms and is capable of dissipating up to 600 Watts of power. The rheostat will simulate a situation where, for example, an accessory motor is continuously engaged due to a failed limit microswitch. This could also represent losses in efficiency in the power distribution system, due to, for example, degradation of switching elements.

3.3.4. Motor Controller Failure

Several types of motor controller faults will be emulated by changing its settings. The simplest is to disable a controller completely, simulating a failure in the system which leaves the wheel unpowered but able to rotate. Another fault can be injected by setting the commanded speed to 0, leaving the wheel to drag (to the limit of the controller's ability). Other types of faults, including the effects of various software errors, will also be investigated and implemented.

3.3.5. Increased Motor Friction

An increased friction fault (caused by a jam in the support bearing or the gearbox, for example) can result in decreased efficiency, increased current consumption, overheating of motor windings, deterioration of their insulation, and eventual failure of the motor due to a short in the windings. To ensure realism, a performance region for the motor will be chosen where a healthy motor would have no problems keeping up with either speed or load requirements. With friction increased, however, the amount of current needed to satisfy the same speed and load demands will be higher, leading to overheating. Unless speed and/or load are reduced or duty cy-

cle (the proportion of time the motor is on versus duration of cool-down intervals) is adjusted, the heat build-up will eventually destroy the insulation of the windings and lead to motor failure. This fault mode was first implemented in the simulator and its model validated using experimental data collected on smaller-sized motors that were run to failure under similar conditions (Balaban, Saxena, Narasimhan, Roychoudhury, & Goebel, 2011). More details on the model validation are provided in Section 4.1. A hardware fault injection using a mechanical brake on one of the rover motors is to be implemented next. The motor will not be run to complete failure initially; instead the model parameters and prognostic algorithms will be validated in experiments stopping just short of creating permanent damage. Eventually, experiments that will take motors all the way to failure will be performed.

3.3.6. Sensor Faults

Sensor faults of particular interest are those that exhibit off-nominal behavior for some time before the failure threshold is reached, such as bias and drift faults. In order to simulate bias and drift faults, constant and increasing offsets will be added to the true measured sensor value, respectively. In addition to bias and drift, other sensor fault modes useful for testing the diagnostic components are 'stuck' (where the sensor value remains unchanged regardless of the system state) and scaling (signal amplification fault).

Sensor faults will be injected by substituting the sensor's value with a different one before sending the data to the reasoning algorithms (Poll et al., 2007; Balaban, Saxena, Bansal, Goebel, & Curran, 2009; Balaban, Saxena, et al., 2011). If the characteristics of a fault can be determined with an acceptable degree of certainty, the decision-making system can attempt to compensate for it by adjusting the output value accordingly (in the case of a drift, bias, or scaling fault) or removing the sensor from the set of active measurements (in the case of a 'stuck' fault).

4. TESTBED SIMULATOR

As previously mentioned, a simulator has been developed to aid in the design of PDM algorithms and use in the decision-making process. It captures both nominal and faulty behavior, with a controlled ability to inject faults. In this way, it serves as a virtual testbed through which algorithms can be initially tested and validated. In this section, we describe the underlying physics model used in the simulator.

4.1. Rover Modeling

The rover consists of a symmetric rigid frame with four independently-driven wheels. Generalized rover coordinates are shown in Figure 2, where the rover pose is given by (x, y, θ). The rover length is denoted by l, the width by b, and the distance from the rover center to each wheel by d.

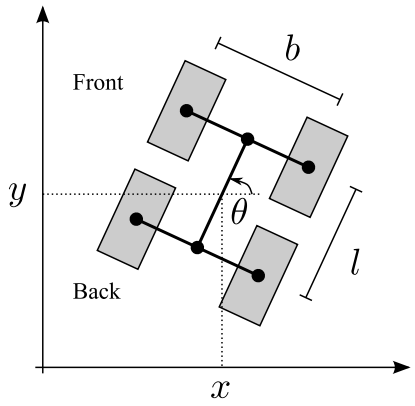


Figure 2. Generalized rover coordinates

4.1.1. Rover Body

Wheels are denoted with F, B, L, and R subscripts, standing for *front*, *left*, *back*, and *right*, respectively, so a wheel w is labeled from the set $\{FL, FR, BL, BR\}$. The forces acting on the rover body are summarized in Figure 3. The slip force on wheel w, F_{sw}, is described by

$$F_{sw} = \mu_s(v_w - v), \tag{1}$$

where μ_s is a friction coefficient, v_w is the translational wheel velocity, and v is the translation velocity of the rover body. When there is a difference between v_w and v, slip is present and the resulting force acts to push the wheel parallel to the ground. When the rover is turning, there are friction forces acting on the wheels opposing the sliding caused by the rotational velocity. The rotational friction force F_{grw} is described by

$$F_{grw} = \mu_{gr} d\omega, \tag{2}$$

where μ_{gr} is a friction coefficient, and ω is the rotational velocity of the rover.

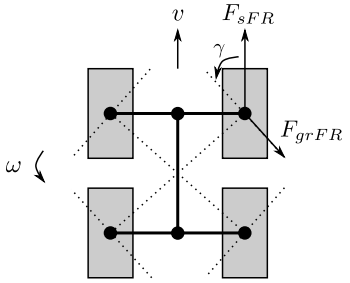


Figure 3. Rover forces

The translational and rotational rover velocities can be expressed based on these forces[1]. The translational velocity v

[1]Note that velocity in the lateral direction is negligible (Mandow et al., 2007).

of the rover can be obtained from

$$\dot{v} = \frac{1}{m}(F_{sFL} + F_{sFR} + F_{sBL} + F_{sBR} + (F_{grFL} - F_{grFR} + F_{grBL} - F_{grBR})\cos\gamma),$$

where m is the rover mass, $\gamma = \arctan\frac{l}{b}$, and the $\cos\gamma$ term projects the F_{grw} forces onto the translational direction. The rotational velocity ω can be obtained from

$$\dot{\omega} = \frac{1}{J}(d(F_{sFR} + F_{sBR} - F_{sFL} - F_{sBL})\cos\gamma - d(F_{grFL} - F_{grFR} - F_{grBL} - F_{grBR})),$$

where J is the rover rotational inertia and the moments are taken in the counter-clockwise direction.

The rotational velocities of the wheels are determined by the torques produced by the ground forces, as well as the motor torque, $\tau_{mw} = k_\tau i_w$, where i_w is the motor current and k_τ is an energy transformation gain; and an axle friction torque that opposes the wheel rotation, $\tau_{fw} = \text{sign}(\tau_{fw_0}) + \mu_{fw}\omega_w$, where τ_{fw_0} is the static friction torque and μ_{fw} is a friction coefficient. In addition, while rolling, due to the deformation of the wheel on the rolling surface, an upward-pointing rolling resistance force is produced ahead of the wheel center, creating a torque that opposes the rotation. The torque depends on speed and can be approximated by $\tau_{rrw} = \text{sign}(\omega_w)\tau_{rrw_0} + k_{rr}\omega_w$ (Genta, 1997), where τ_{rrw_0} is a static friction torque and k_{rr} is an empirical parameter. The wheel speeds are thus governed by

$$\dot{\omega}_{FL} = \frac{1}{J_m}\left(\tau_{mFL} - \tau_{fFL} - \tau_{sFL} - \tau_{rrFL} + \tau_{grFL}\right), \tag{3}$$

$$\dot{\omega}_{FR} = \frac{1}{J_m}\left(\tau_{mFR} - \tau_{fFR} - \tau_{sFR} - \tau_{rrFR} - \tau_{grFR}\right), \tag{4}$$

$$\dot{\omega}_{BL} = \frac{1}{J_m}\left(\tau_{mBL} - \tau_{fBL} - \tau_{sBL} - \tau_{rrBL} + \tau_{grBL}\right), \tag{5}$$

$$\dot{\omega}_{BR} = \frac{1}{J_m}\left(\tau_{mBR} - \tau_{fBR} - \tau_{sFR} - \tau_{rrBR} - \tau_{grBR}\right), \tag{6}$$

where J_m denotes the wheel/motor inertia; $\tau_{sw} = r_w F_{sw}$ is the torque due to slippage, where r_w is the wheel radius; $\tau_{gtw} = r_w F_{gtw}$ is the torque due to the translational rover friction force; and $\tau_{grw} = r_w F_{grw}\cos\gamma$ is the torque due to the rotational rover friction force.

The rover pose is then described by

$$\dot{x} = v\cos\theta, \tag{7}$$

$$\dot{y} = v\sin\theta, \tag{8}$$

$$\dot{\theta} = \omega. \tag{9}$$

4.1.2. Motors

The wheels are driven by direct-current (DC) motors with proportional-integral-derivative (PID) control that sets the voltages V applied to the motors as a pulse-width modulated (PWM) signal. Here, we ignore the PWM dynamics and assume an averaged model. For wheel w, the motor currents i_w are governed by

$$\frac{di_w}{dt} = \frac{1}{L}(V_w - i_w R_w - k_\omega\omega_w). \tag{10}$$

Here, L is the motor inductance, R is the motor resistance, and k_ω is an energy transformation term. The voltages applied to the motors are determined by the controllers, where for wheel w, $V_w = P\cdot(u_w - \omega_w) + I\cdot e_{iw} + D\cdot e_{dw}$, where P is a proportional gain, u_w is the commanded wheel speed, I is an integral gain, e_{iw} is the integral error term, D is a derivative gain, and e_{dw} is the derivative error term.

The motor controllers take in total battery voltage V_B and step it down to apply to the motors to control the speed. The currents drawn from the batteries by each motor controller, i_{bw} for wheel w, respect a power balance with some loss due to motor controller power efficiency. For wheel w, these currents are therefore determined as

$$i_{bw} = \frac{\eta_w V_w i_w}{V_B}, \tag{11}$$

where η_w is the motor controller power efficiency.

The motor windings are designed to withstand temperatures up to a certain point, after which, the insulation breaks down, the windings short, and the motor fails. It is therefore important to model the temperature behavior of the motor. The motor thermocouple is located on the motor surface. The surface loses heat to the environment and is heated indirectly by the windings, which, in turn, are heated up by the current passing through them. The temperature of the windings for the motor of wheel w is given by

$$\dot{T}_{dw} = \frac{1}{C_{dw}}\left(i_w^2 R - h_{dw}(T_{dw} - T_{mw})\right), \tag{12}$$

where C_{dw} is the thermal capacitance of the windings, h_{dw} is a heat transfer coefficient, and T_{sw} is the motor surface temperature (Balaban, Narasimhan, et al., 2011). It is assumed that heat is lost only to the motor surface, and that winding resistance R is approximately constant for the temperature range considered. The surface temperature of the motor for wheel w is given by

$$\dot{T}_{sw} = \frac{1}{C_{sw}}(h_{dw}(T_{dw} - T_{sw}) - h_{aw}(T_{sw} - T_a)), \quad (13)$$

where C_{sw} is the thermal capacitance of the motor surface, h_{aw} is a heat transfer coefficient, and T_a is the ambient temperature. Heat is transferred from the windings to the surface and lost to the environment.

The motor temperature model was validated for DC motors using experimental data collected on the Flyable Electro-Mechanical Actuator (FLEA) testbed (Balaban et al., 2010). The unknown parameters C_{dw}, C_{sw}, h_{dw}, h_{aw}, and R were identified to match data acquired from a scenario where the motor was overloaded and, as a result, heated up considerably. The motor current and surface temperatures were measured. A comparison of predicted vs. measured temperature is shown in Figure 4.

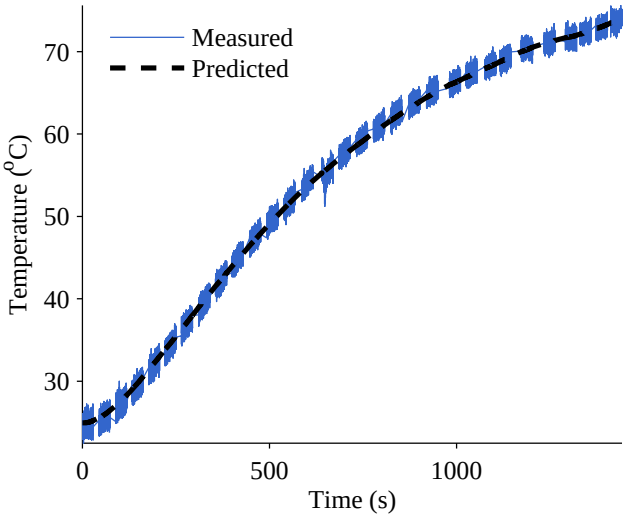


Figure 4. Comparison of measured and model-predicted motor surface temperature for a DC motor

4.1.3. Batteries

The input voltage to the motors is provided by the rover batteries. The battery model is based on an electrical circuit equivalent shown in Figure 5, and extends the model presented in (Daigle, Saxena, & Goebel, 2012) that is similar to models presented in (Chen & Rincon-Mora, 2006; Ceraolo, 2000). The large capacitance C_b holds the charge q_b of the battery. The R_{CP}-C_{CP} pair captures the major nonlinear voltage drop due to concentration polarization, R_s-C_s pair captures the so-called I-R drop, and R_p models the parasitic resistance that accounts for self-discharge. This simple battery model is enough to capture the major dynamics of the battery, but ignores temperature effects and other minor battery processes.

The state-of-charge, SOC, is computed as

$$SOC = 1 - \frac{q_{max} - q_b}{C_{max}}, \quad (14)$$

where q_b is the current charge in the battery (related to C_b), q_{max} is the maximum possible charge, and C_{max} is the maximum possible capacity. The concentration polarization resistance is a nonlinear function of SOC:

$$R_{CP} = R_{CP0} + R_{CP1} \exp\left(R_{CP2}(1 - SOC)\right), \quad (15)$$

where R_{CP0}, R_{CP1}, and R_{CP2} are empirical parameters. The resistance, and, hence, the voltage drop, increases exponentially as SOC decreases (Saha et al., 2012).

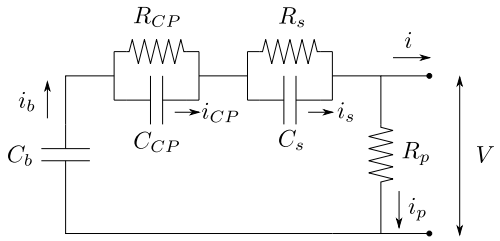


Figure 5. Battery equivalent circuit

Voltage drops across the individual circuit elements are given by

$$V_b = \frac{q_b}{C_b}, \quad (16)$$

$$V_{CP} = \frac{q_{CP}}{C_{CP}}, \quad (17)$$

$$V_s = \frac{q_s}{C_s}, \quad (18)$$

$$V_p = V_b - V_{CP} - V_s, \quad (19)$$

where q_{CP} is the charge associated with the capacitance C_{CP}, and q_s is the charge associated with C_s. The voltage V_b is also the open-circuit voltage of the battery, which is a nonlinear function of SOC (Chen & Rincon-Mora, 2006). This is captured by expressing C_b as a third-order polynomial function of SOC. The terminal voltage of the battery is

$$V = V_b - V_{CP} - V_s. \quad (20)$$

Currents associated with the individual circuit elements are given by

$$i_p = \frac{V_p}{R_p}, \quad (21)$$

$$i_b = i_p + i, \quad (22)$$

$$i_{CP} = i_b - \frac{V_{CP}}{R_{CP}}, \quad (23)$$

$$i_s = i_b - \frac{V_s}{R_s}, \quad (24)$$

where $i = i_{bFL} + i_{bFR} + i_{bBL} + i_{bBR}$ is the battery current at the terminals.

The charges are then governed by

$$\dot{q}_b = -i_b, \tag{25}$$
$$\dot{q}_{CP} = i_{CP}, \tag{26}$$
$$\dot{q}_s = i_s. \tag{27}$$

The battery temperatures are modeled with a simple thermal model described as follows. For battery k, the temperature is described by

$$\dot{T}_{bk} = \frac{1}{C_{btk}} \left(R_{btk} i_b^2 + h_{btk}(T_a - T_{bk}) \right), \tag{28}$$

where C_{btk} is a thermal capacitance, R_{btk} is a thermal resistance, i_b is the total battery current, and h_{btk} is a heat transfer coefficient.

Two parallel sets of batteries power the rover. Each set consists of 12 cells connected in series to provide a nominal of 48 V. When fully charged, each cell provides up to 4.2 V. The current i is split between the two sets, so each individual battery is drained by a current of $i/2$ when balanced.

The battery model was validated with data from the battery testbed at NASA Ames Research Center (Saha & Goebel, 2009). A comparison of real and measured voltage for a constant discharge is shown in Figure 6, where it is clear that the model predicts a nominal discharge curve very accurately.

4.1.4. Sensors

Sensors measure the voltages and temperatures of the batteries (V_{bi} and T_{bi}), the motor currents (i_{bw}), the total current (i), the motor temperatures (T_{mw}), the wheel positions, and the wheel speeds (ω_w).

The phone provides additional GPS, accelerometer, gyroscope, and magnetometer sensors. The nominal output of the GPS sensor is modeled as

$$\lambda = \lambda_0 + \arctan\left(\frac{y}{r_E}\right), \tag{29}$$
$$\phi = \phi_0 + \arctan\left(\frac{x}{r_E}\right), \tag{30}$$
$$h = h_0, \tag{31}$$

where λ_0 is the initial latitude, ϕ_0 is the initial longitude, and r_E is the radius of the Earth. Here, we assume that the rover does not undergo elevation changes within the measurement interval ($\Delta h = 0$) and that the curvature of the Earth is approximately flat within the rover's operating range. The phone also provides accelerometer, gyroscope, and magnetometer measurements. The accelerometer readings in the x, y, and z directions are computed as

$$a_x = \dot{v}\cos(\theta) + \dot{w}\frac{l}{2}\cos\left(\theta + \frac{\pi}{2}\right), \tag{32}$$
$$a_y = \dot{v}\sin(\theta) + \dot{w}\frac{l}{2}\sin\left(\theta + \frac{\pi}{2}\right), \tag{33}$$
$$a_z = 9.81. \tag{34}$$

For the gyroscope, we assume that the only rotation is along the z-axis, so

$$g_x = 0, \tag{35}$$
$$g_y = 0, \tag{36}$$
$$g_z = \omega. \tag{37}$$

For the magnetometer, since we assume only rotation about the z-axis, we have

$$m_x = M\sin(\theta + \theta_d)\cdot\cos\theta_i, \tag{38}$$
$$m_y = M\cos(\theta + \theta_d)\cdot\cos\theta_i, \tag{39}$$
$$m_z = M\cdot\sin\theta_i, \tag{40}$$

where M is the maximum output of the magnetometer in any one axis for the area of operation, θ_d is the declination angle from true north (the angle between magnetic and true north at the rover location), and θ_i is the inclination angle from horizontal.

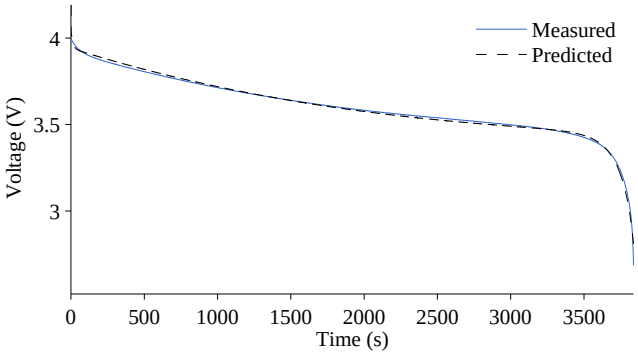


Figure 6. Comparison of measured and model-predicted battery voltage for a single battery cell

4.2. Fault Modeling

Most faults are captured in the model as changes in parameter values. For example, increased motor friction is represented through changes in the friction coefficients (μ_{fw}), and internal resistance increases in the batteries associated with aging are represented through changes in R_s and R_{CP}. A parasitic load on the batteries is captured as an additional current, i_{pl}, that is drawn from the batteries. Sensor faults are captured with bias, drift, and gain terms. Ranges for typical fault magnitude values have been identified through a literature search and discussions with manufacturers (Balaban et al., 2009). Faults in common sensors, such as current, voltage, temperature, and position, are modeled.

4.3. End of Life Constraints

We are interested in predicting when any of the rover battery cells are at their voltage threshold, beyond which the cells will be damaged. The constraints are given as

$$c_i : V_i > V^-, i \in \{1, 2, 3, 4\}, \tag{41}$$

where the voltage threshold is given by $V^- = 3.0$ V.

We are also interested in when the motor temperature gets too high. The motors windings are designed to withstand temperatures up to a certain point, after which, the insulation breaks down, the windings short, and the motor fails (see subsection 3.3). The constraints are given as

$$c_5 : T_{dFL} < T_d^+, \tag{42}$$

$$c_6 : T_{dFR} < T_d^+, \tag{43}$$

$$c_7 : T_{dBL} < T_d^+, \tag{44}$$

$$c_8 : T_{dBR} < T_d^+, \tag{45}$$

where the temperature limit is given by $T_d^+ = 70^\circ$ C.

The rover cannot be operated when one or more of these constraints are violated.

5. INTEGRATED ARCHITECTURE

A typical rover mission consists of visiting and performing desired science functions at a set of predetermined waypoints $W = \{(x_i, y_i), r_i\}_{i=1}^N$, where r_i is the reward associated with location (x_i, y_i). An autonomous PDM system for the rover determines the order to visit the waypoints and the speed v_i to travel between them, so as to maximize the reward while minimizing the power used and health deterioration. When diagnostic and prognostic information is available, the decision making system may alter the waypoint list (reduce and/or reorder) to minimize the impact of the faults and slow their progression over time. Figure 7 presents the integrated decision making architecture, which consists of four main components, namely (*i*) locomotion controller (LC), (*ii*) diagnostics (DX), (*iii*) prognostics (PX), and (*iv*) decision making (DM).

The LC is responsible for guiding the rover to the current waypoint, $w_i = \{\{x_i, y_i\}, r_i\}$, by commanding individual wheel velocities to be v_t at time, t. The rover is a skid-steered vehicle, meaning that the wheels cannot be steered and the rover is rotated by commanding the wheel speeds on the left and right sides to different values. The low-level controller must be robust to drive system faults, therefore, it receives the current diagnosis $\mathbf{F}_t$ from the DX algorithm.

The DX module takes in the inputs $\mathbf{u}_t$ and sensor data $\mathbf{z}_t$, and reasons about faults in the system. The DX runs continuously, trying to detect when a fault (or faults) occur, isolate the actual fault(s) from a set of possible candidates, and identify fault magnitudes. In case of a fault, as a diagnosis $\mathbf{F}_t$ becomes available, it is passed on to the LC for mitigation, as well as to the prognoser, which uses the information as a starting point for predicting fault progression.

If an off-nominal condition is detected, the PX module begins to continuously estimate the health state of the affected components (and the rover overall) and generates predictions of the remaining useful life, $\mathbf{RUL}_t$. RUL is the time until the system violates functional or performance constraints. The RUL prediction is conditional on the future usage of the vehicle, determined by the waypoint list.

The DM module is responsible for planning under nominal and faulty conditions. At the mission start, the DM gets a set W of desired waypoints. The DM also has a terrain map $\mathcal{M}$ that describes the geographical layout of the terrain on which the waypoints are located. The DM then, based on the predicted power usage and RUL predictions, determines an ordered (and possibly reduced) list of waypoints. A number of different protocols may be implemented for invoking the module. For example, after the first solution is produced, the DM may be called again if PX predictions start diverging substantially from the initial ones. Alternatively, DM may be called after each waypoint is reached.

We anticipate that the architecture described in this section can be applied with only minimal changes to an aerial test vehicle, such as the Edge 540 unmanned aerial vehicle described in (Hogge, Quach, Vazquez, & Hill, 2011). Motion control components would, of course, need to be adapted to operate with six degrees of freedom instead of three. Execution of the reasoning algorithms would need to be optimized to accommodate the faster position change rates and to account for the fact that an aerial vehicle, unlike a rover, does not have the option of pausing motion while a fault is being analyzed or a new course of action is being formulated.

6. LOCOMOTION CONTROL

The task of the locomotion controller is to direct the rover to a given waypoint $w_i = \{(x_i, y_i), r_i\}$ by sending appropriate wheel speed commands to the individual motors based on the estimated pose $(\hat{x}, \hat{y}, \hat{\theta})$. We define two different control strategies: one which does not take into account diagnostic information and one that does.

6.1. Proportional Control

The waypoint w_i and desired cruise speed u_v are given as inputs to the proportional controller. It implements a proportional control based on the error between the current estimated heading $\hat{\theta}$ and the desired heading θ^*, e_θ. The average wheel speed is set to u_v and the desired speeds of the left and right sides are set to turn toward the waypoint based on e_θ:

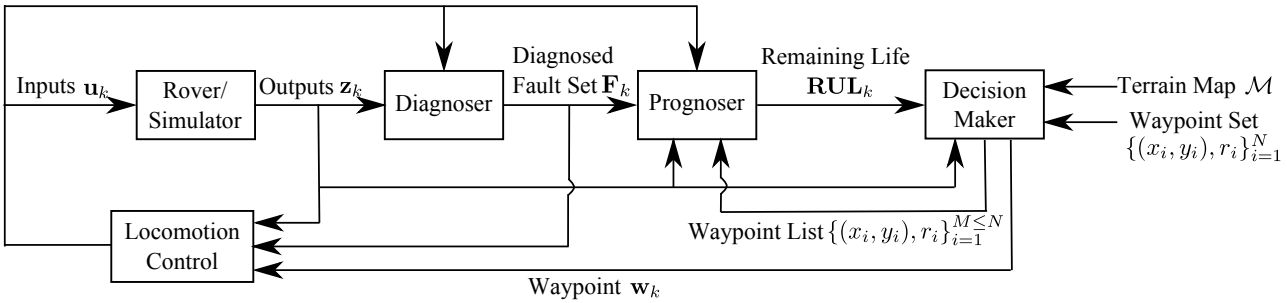


Figure 7. Autonomy Architecture

$$u_L = u_v - P_\theta e_\theta, \tag{46}$$
$$u_R = u_v + P_\theta e_\theta, \tag{47}$$

where P_θ is the proportional gain. The individual wheel speeds are then set to

$$u_{FL} = u_L, \tag{48}$$
$$u_{FR} = u_R, \tag{49}$$
$$u_{BL} = u_L, \tag{50}$$
$$u_{BR} = u_R. \tag{51}$$

6.2. Differential Speed

In the proportional control, the wheel speeds on one side of the rover are always set to the same value. However, if a motor friction fault is present on one side of the rover, then it is more efficient overall to increase the desired velocity of the good wheel and decrease the velocity of the faulty wheel on that side. The optimal amount of change depends on the magnitude of the friction fault.

This controller uses the proportional control as the basis and modifies its outputs based on the value of the friction coefficient. First it computes nominal wheel speeds for the left and right sides (equations 46-47). Then, it modifies the speeds commanded to the front and back wheels on each side according to the estimated friction coefficient.

$$u_{FL} = u_L \frac{2}{1 + \frac{1+\mu_{fFL}}{1+\mu_{fBL}}}, \tag{52}$$
$$u_{FR} = u_R \frac{2}{1 + \frac{1+\mu_{fFR}}{1+\mu_{fBR}}}, \tag{53}$$
$$u_{BL} = u_L \frac{2}{1 + \frac{1+\mu_{fBL}}{1+\mu_{fFL}}}, \tag{54}$$
$$u_{BR} = u_R \frac{2}{1 + \frac{1+\mu_{fBR}}{1+\mu_{fFR}}}. \tag{55}$$

Note, from equations 52 and 54, that

$$\frac{u_{FL} + u_{BL}}{2} = u_L. \tag{56}$$

Similarly, from equations 53 and 55

$$\frac{u_{FR} + u_{BR}}{2} = u_R. \tag{57}$$

On a given side of the rover, if the friction on the front wheel is larger than friction on the back wheel, the speed of the front wheel is decreased and the speed of the back wheel is increased. If the back wheel has more friction than the front wheel, its speed is decreased while the speed on the front wheel is increased. The overall speed on that side of the rover remains the same, but the total current drawn is less than if both wheels were set to the same speed.

7. REASONING ALGORITHMS

The primary purpose for building the K11 is to enable hardware-in-the-loop testing of reasoning algorithms (diagnostic, prognostic, and decision-making). The current set of algorithms, in addition to demonstrating certain novel PDM ideas, serves to validate the integrated reasoning architecture and the concept of operations for the testbed. Some of the algorithms have been used in previous research efforts (e.g. Qualitative Event-based Diagnosis (QED), Hybrid Diagnostic Engine (HyDE), and some of the prognostic methods), while others are being developed concurrently with the K11. The details of the algorithms are provided in the rest of this section.

7.1. Diagnosis

Diagnosis involves *detection*, *isolation*, and *identification* of faults. A fault is a change in the system that causes its behavior to deviate from nominal. Detection involves determining when a fault occurs based on some observable symptoms. Isolation involves determining the true fault out of a set of, possibly, several fault candidates, and identification is the process of determining the extent of the damage to the system. Numerous diagnosis approaches exist in the literature. We apply two model-based diagnosis approaches to the

rover testbed: the QED algorithm and the HyDE, both developed at NASA Ames Research Center on the basis of earlier work at Vanderbilt University (the algorithms are overviewed in the following subsections). The motivation for two distinct algorithms to be used in the same role is to demonstrate the ability of the testbed to easily accommodate reasoning algorithms from different sources, as long as they adhere to the same interfaces. Several diagnostic algorithms could even, potentially, run in parallel, with their output aggregated by a higher level reasoner.

7.1.1. QED

QED, described in (Daigle & Roychoudhury, 2010), utilizes a qualitative diagnosis methodology that isolates faults based on the transients they cause in system behavior, manifesting as deviations in residual values (Mosterman & Biswas, 1999). Transients produced by faults are abstracted using qualitative + (increase), − (decrease), and 0 (no change) values to form *fault signatures*. Fault signatures represent these measurement deviations from nominal behavior as the immediate (discontinuous) change in magnitude and the first nonzero derivative change. These symbols are computed from the residuals using symbol generation. In addition to signatures, QED captures the temporal order of measurement deviations, termed *relative measurement orderings*. The fault signatures and measurement orderings can be computed manually or automatically from a system model. They are compared with observed signatures and orderings in order to isolate faults. The combination of signatures and orderings establishes an event-based fault isolation framework.

QED uses the model of the rover described in Section 4. Given this model, fault signatures and measurement orderings are derived. Algebraic functions computing fault magnitudes are also derived and used for fault identification.

7.1.2. HyDE

HyDE is a consistency-based diagnosis engine that uses hybrid (discrete/continuous) models and reasoning (Narasimhan & Browston, 2007). The models are also allowed to incorporate stochastic behavior. Users first build models of constituent components of the system and then compose the system model by defining the connections between the components. Component models include a set of discrete modes (nominal or off-nominal) the components could be in and the behavior of the component in each mode. A transition from a nominal mode to a fault mode indicates an occurrence of the corresponding fault. At any point in time, HyDE maintains a set of candidates that offer alternative versions of the system state that are consistent with the sensor observations seen so far (the size of the candidate set can be limited). When additional observations become available, the candidate set is pruned of inconsistent candidates and augmented with new consistent candidates. Several parameters are available to adjust the performance of the engine, including heuristics to determine the candidate ranking or the type of system simulation used.

Similarly to QED, the HyDE model for the rover was developed directly from the simulation models, with the components and equations in one-to-one correspondence. Fault modes were defined for all of the sensors, as well as for the motors (increased friction fault) and batteries (parasitic load fault).

7.2. Prognosis

Prognosis is concerned with predicting the end of (useful) life (EOL) and RUL of a component, subsystem, or system. EOL is defined as the earliest time point at which the system no longer meets specified functional or performance constraints, and RUL is the time remaining until that point. These constraints do not necessarily have to correspond to complete failure, e.g., for the rover we are interested in when a battery reaches end-of-discharge. In order to predict EOL/RUL, we require an estimate of the current system state (including known fault conditions), some estimate of the future usage of the system, and some model that can predict the evolution of the system state up to EOL. This model may be determined through physical modeling of the system (Daigle, Saha, & Goebel, 2012) or through data-driven methods (Schwabacher, 2005).

For the rover, prognostics algorithms can be used for several components. For the batteries, we must predict when a battery will be fully discharged, because the rover cannot be operated beyond that point. This has obvious implications for decision-making. As the batteries are used over several rover missions, they will naturally age and the capacity rating will drop, therefore we must also predict when the battery capacity falls below the required minimum. For the drive motors, we need to predict when an overheating may occur, since it can lead to permanent motor damage.

7.2.1. Model-based Prognosis

The model-based prognosis paradigm (Orchard, 2007; Daigle, Saha, & Goebel, 2012) consists of two steps: (*i*) state estimation, which computes a joint state-parameter estimate of the system, and (*ii*) prediction, which simulates the model forward from a given health state out to the EOL threshold, based on hypothesized future inputs to the system. State estimation can be performed with Kalman filters (Celaya, Kulkarni, Biswas, & K., 2012), unscented Kalman filters (Daigle, Saha, & Goebel, 2012), particle filters (Orchard, 2007; Saha & Goebel, 2009; Daigle & Goebel, 2011), or similar algorithms. Prediction is typically performed by sampling from the state estimate and simulating each sample to EOL, taking into account process noise and future input uncertainty

(Daigle, Saxena, & Goebel, 2012).

The prediction step requires knowledge of the future usage of the system. For the rover, this involves the expected future trajectory and environmental inputs, such as the terrain and the ambient temperature. The physics models developed for the simulator can be utilized in both the estimation and prediction phases. Damage progression processes that are difficult to model may require the use of data-driven prognostics methods.

As an example of the model-based prognosis approach, consider prognostics of the rover batteries. In this particular case, we predict when the battery voltage will decrease to 3 V (focusing on one particular battery). The total current drawn from the batteries is shown in Figure 8a (recall that each individual battery sees only half that current). The measured and estimated voltages are shown in Figure 8b. The battery does not start at full charge (which would be approximately 4.2 V) and so reaches EOL around 1800 s. The estimated battery SOC is shown in Figure 8c, and, according to the model, the battery starts at only 22% SOC. The corresponding prediction results, assuming future inputs known exactly, are shown in Figure 8d. As can be seen, the predictions are very accurate from early on in the process, which means that, since the future inputs are known, the model is accurate in the open loop.

7.2.2. Data-driven Prognosis

In data-driven prognostics the degradation model for a fault mode is learned from training data. While potentially requiring less domain knowledge *a priori* than model-based techniques, reliance on availability of such data can be a limiting factor in some types of practical applications. Methods such as Neural Networks, Relevance Vector Machines, and Gaussian Process Regression (GPR) have been utilized for data-driven prognostics (Schwabacher, 2005).

In this work a machine-learning algorithm based on GPR is used to predict when the interior of a drive motor would reach the temperature at which insulation of the windings is likely to melt, thus disabling the motor. This fault mode is likely to occur if the motor, in addition to its nominal load, has to compensate for increased mechanical friction. An increase in friction can occur due, for example, to a failed bearing or a damaged gear inside the gearbox (please refer back to subsection 3.3 for more details). The thermal build-up model, as described in the Section 4, will be used for comparison.

The GPR-based algorithm was previously tested in prognostic experiments involving motor faults in electro-mechanical actuators (Balaban, Saxena, et al., 2011). An increased friction fault was injected and the relevant sensor data collected as the actuator continued to be used in a degraded state (various position and load profiles were utilized in the experiments). Several motors were run to complete failure, with GPR demonstrating a high accuracy in predicting their remaining useful life.

GPR does not need explicit fault growth models and can be made computationally less expensive by sampling techniques. Further, it provides variance bounds around the predicted trajectory to represent the associated confidence (Rasmussen & Williams, 2006). Domain knowledge available from the process is encoded by the covariance function that defines the relationship between data points in a time series. In our present implementation, a Neural Network type covariance function is used.

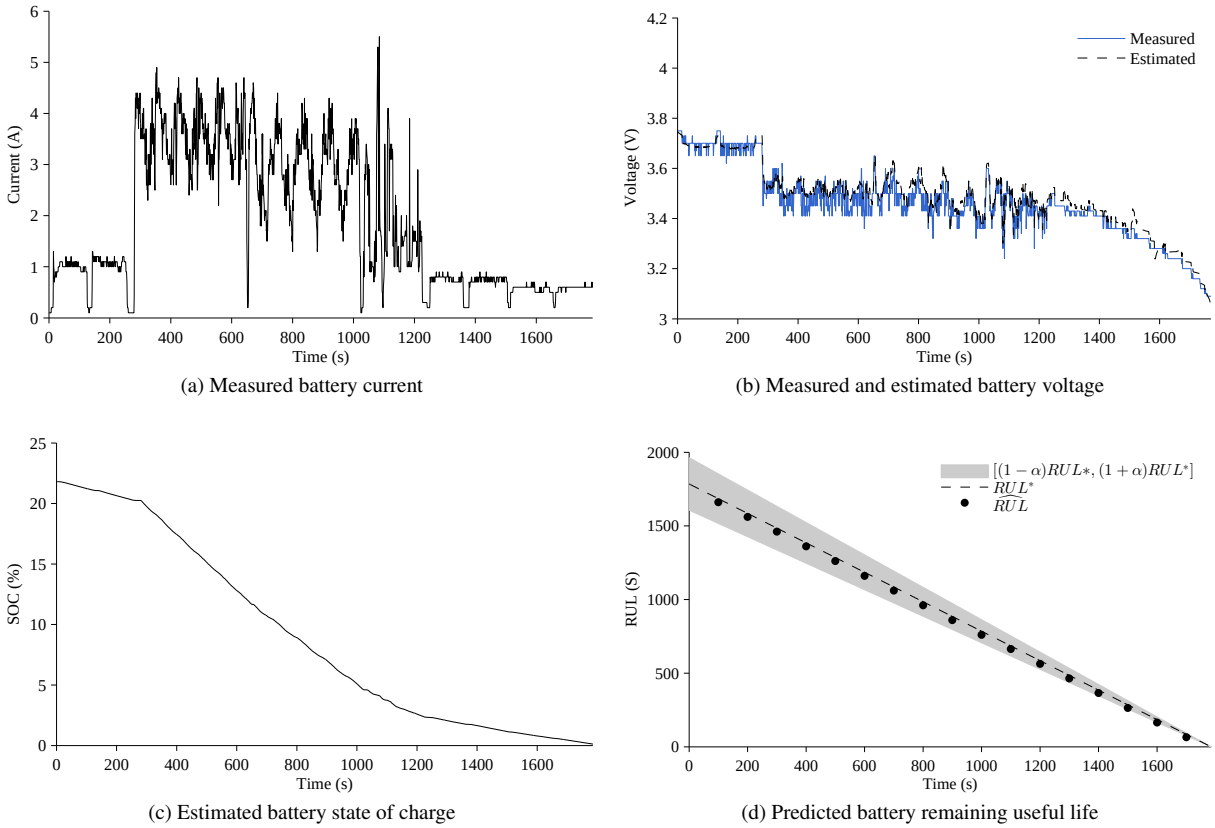
7.3. Decision Making

The overall objective of this work is to develop PDM algorithms that enhance a vehicle's capability to achieve its high-level goals - be it under a localized fault condition, degraded operation of a subsystem, or an anticipated catastrophic failure. There has been an increasing volume of research conducted over the last several years in prognostic methodologies for various types of components or systems. The effort described in this section aims to bring more attention to the management aspect of prognostic health management, i.e. what could be done after a fault is detected and a prognostic prediction for it is produced.

While the components involved and the particulars of each off-nominal situation may be different, there are, however, common elements in all of them. There is always an objective (or a set of objectives) to be met and a series of actions to be selected by the decision making process in order the meet those objectives. In many circumstances a satisfactory solution (one that satisfies constraints defined for the system) is all that is required; in this work, however, we chose to define the decision-making process as an optimization problem, where a Pareto Optimal Set of solutions (Fudenberg & Tirole, 1991) is produced given the objectives, prognostic system estimates, system constraints, and the projected future operating conditions. The Pareto formulation is used to accommodate multiple potential objectives.

Some of our initial work in PDM was described in (Balaban, Narasimhan, et al., 2011). A more recent publication (Balaban & Alonso, 2012) provides a PDM problem formulation from a multi-objective optimization point of view and describes the overall approach being pursued. The current work focuses on the following aspects of PDM: (*i*) decomposition of the overall decision-making problem into smaller subproblems; (*ii*) subproblem modeling methods; (*iii*) subproblem decision-making algorithms; and (*iv*) methods for adjusting problem formulations (such as constraints or objectives) in real-time, if necessitated by prognostic predictions in off-nominal situations.

Two K11-related case studies are used in research in these



(a) Measured battery current

(b) Measured and estimated battery voltage

(c) Estimated battery state of charge

(d) Predicted battery remaining useful life

Figure 8. Battery prognosis results

four areas. One involves relaxation of the maximum operating temperature constraints on the batteries and motors in order to achieve a desired destination in the presence of an increased friction fault. The other is built around the waypoints optimization scenario mentioned in the earlier parts of the paper and described in detail in Section 8.

For the constraint relaxation case study, both a method to perform system decomposition from the game-theoretic perspective and an approach to adjust select system constraints (if that is needed to satisfy high-priority performance goals) are explored. Decomposition is performed based on the subsystem-level abstraction, where the subsystems cooperate in exploring the (potentially) very large option space by taking turns in searching for a suitable solution. The current formulation of the decomposition algorithm tests the concept for two rover subsystems (power and propulsion), with extension to larger numbers of subsystems planned for subsequent versions.

The waypoints optimization case study forms the basis of the experiments covered in this paper. It motivated the work of exploring decision-making models suitable for PDM and algorithms capable of using the models to generate an optimal action policy in the presence of system degradation, multiple objectives, uncertainty, and constraints. The general modeling approach currently used for waypoints optimization is Partially Observable Markov Decision Process, POMDP (Cassandra, Kaelbling, & Littman, 1995). Several algorithms have been applied to generate solutions. Dynamic Programming, backtracking search, and Particle Filter are among them. In the set of experiments presented in the next section a stochastic algorithm patterned on the Probability Collectives ($\mathcal{PC}$) approach (Wolpert, 2006) is used. The algorithm belongs to the class of blackbox optimization methods. Such methods, generally, have the goal of finding a value $x \in X$ that minimizes an associated value $F(x)$. X is an optimization space and $F(x)$ could be an objective or a utility function. The following process is repeated iteratively: (1) an x is chosen from X; (2) statistical information about $F(x)$ is updated; (3) the next value of x is chosen using the $(x, F(x))$ pairs found up to that point. For waypoint optimization, partial paths form the optimization space X and objective functions are created for scientific payoff, energy use, and vehicle health. Further details on the approach and the algorithm implemented are provided in (Balaban & Alonso, 2012).

8. EXPERIMENTS AND RESULTS

This section provides examples of integrated prognostics-enabled decision making on a set of fault scenarios executed on the K11 simulator. The case study presented is that of the rover starting its mission at a waypoint w_0 and attempting to visit, at an average speed of 0.5 m/s, a set of other 10 waypoints, accomplishing a scientific objective at each. As mentioned previously, every waypoint is associated with a reward value and the primary objective is to maximize the cumulative reward. In absence of system faults the rover has enough energy stored in its batteries to visit all of the waypoints. In addition to this nominal scenario, however, three fault scenarios are considered: a parasitic load in the power distribution system, increased motor friction (Figure 9), and a battery voltage sensor fault. In the fault scenarios, the diagnostic system is expected to detect and identify the fault mode during the $w_0 \rightarrow w_1$ transition. The decision-making system is then expected to modify the waypoint traversal plan, taking into account prognostic estimates of future energy consumption and fault magnitude progression. An *a posteriori* estimate of change in these two quantities during the $w_0 \rightarrow w_1$ transition, given the fault diagnosis, is made as well.

8.1. Battery Parasitic Load Fault

As a first scenario, we consider an abrupt parasitic load fault that draws an additional amount of current from the batteries. A parasitic current of 0.1 A is injected starting at 50 s. As a result, the net current increases and the battery voltages decrease in response to the increased current. No other effects appear, but the batteries will drain faster and therefore path analysis will have to be performed to determine if all of the waypoints can still be visited with this increased load. See Figure 10 for a diagram that illustrates the timing of various events in this scenario.

QED detects the fault at 50.4 s with an increase in i, the current drawn from the batteries. The candidate set reduces to: motor failures, motor friction faults, the parasitic load fault, and a bias or drift in the current sensor. At 51.2 s, the change is determined to be abrupt, eliminating the friction faults and the drift fault. At 76.15 s, a decrease in the voltage of battery 4 is detected. This eliminates the current sensor fault, since it would not affect another sensor, and eliminates the motor failure faults, as these would cause a change in individual motor currents before a change in voltage (i.e., there is a relative measurement ordering for these faults expressing this constraint), thus uniquely isolating the parasitic load fault. The parasitic current is computed as the difference between the battery current and the sum of the motor currents, averaged from the time of detection to the present time. At 51.2 s, the value of the parasitic current is estimated at 0.127 A. By 100 s, the estimate is at 0.103 A.

HyDE detects the fault at 50.05 and isolates the fault to either a parasitic load or one of the motors being in an unknown mode.[2] In addition, HyDE was tested on a double fault scenario where there is a battery parasitic load and the battery current sensor is faulty. HyDE is able to use other sensors to determine this condition. The overall result is the same as with the parasitic load fault, since sensor faults do not affect the decision-making.

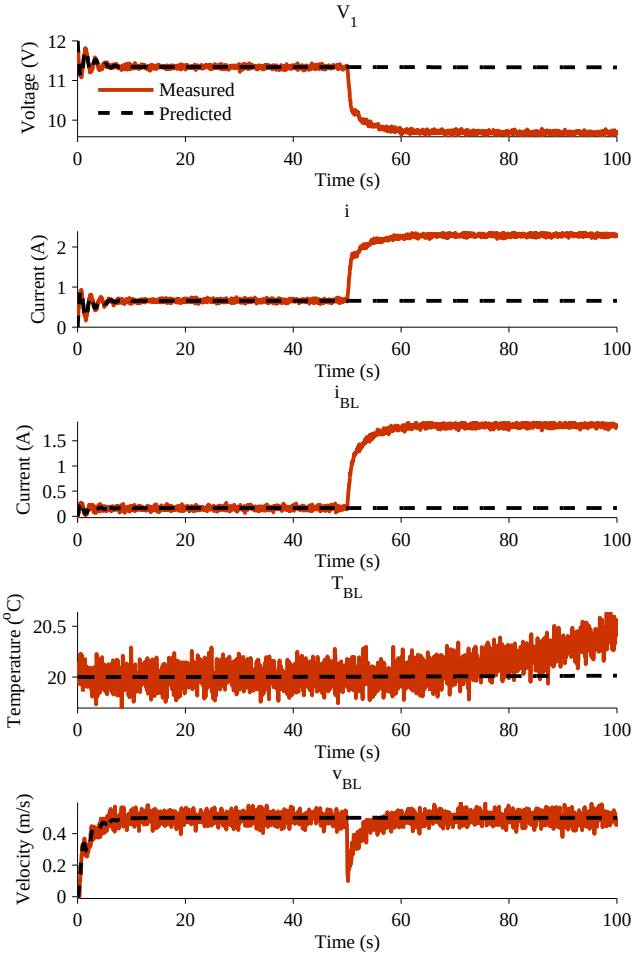


Figure 9. Sensor readings for motor friction fault

Prognosis begins once the fault is identified. Here, we determine when the batteries will reach end-of-discharge with the current mission plan. The prognosis algorithm reports that the batteries will reach this point in 3.83 h (as opposed to 4.54 h without the fault). Continuing at an average speed of 0.5 m/s, the rover can cover about 6.9 km with the fault (8.2 without the fault). Since the rover must cover at least 6.9 km to visit all of the waypoints, mission optimization is required. The DM module estimates that there will not be

[2] In the unknown mode, the behavior of the motor is undefined and hence it could be drawing more current. However, unknown faults are catch-all modes that should be considered only when no other explanations are available.

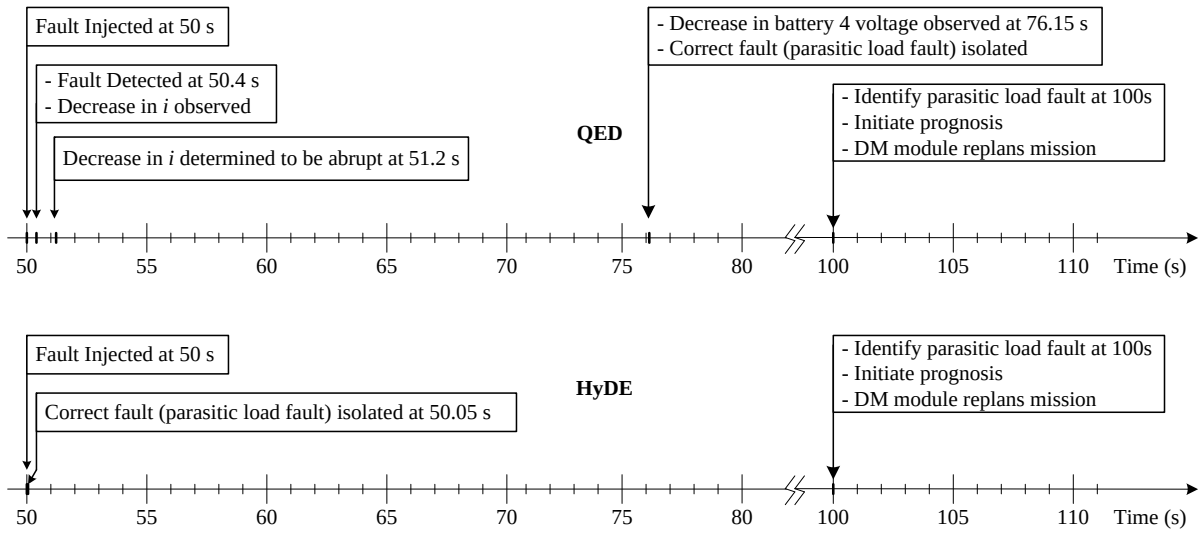


Figure 10. Timing diagram for the battery parasitic load fault scenario

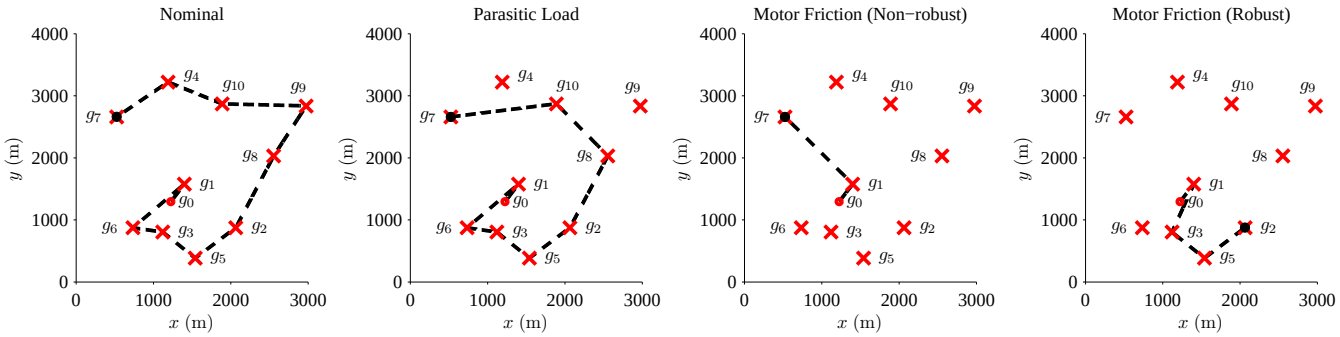


Figure 11. Decision making results (with $r_1 = 40$, $r_2 = 50$, $r_3 = 30$, $r_4 = 20$, $r_5 = 60$, $r_6 = 40$, $r_7 = 100$, $r_8 = 50$, $r_9 = 30$, and $r_{10} = 60$)

enough energy to visit all of the waypoints and eliminates waypoints 9 and 4 from the plan, reconfiguring the path to be $q_s = \{1, 6, 3, 5, 2, 8, 10, 7\}$. See Figure 11 for the decision-making results under different scenarios.

8.2. Motor Friction Fault

As a second scenario, we consider a motor friction fault in the back-left motor (Figure 9). The friction coefficient value is increased by a factor of 10 at 50 s. As a result, the motor current increases because its PID controller is still trying to maintain the wheel speed at the same level. This corresponds to an increase in the total current drawn from the batteries and an accelerated rate of discharge (and, thus, decreased RUL). The motor temperature also rises due to the increased current draw, and this can lead to an EOL condition of the motor due to the overheating. As a result, decision making will have to optimize RUL with respect to battery life and motor health.

QED detects the fault at 50.15 s with an increase in v_{BL}, the velocity of the back-left wheel 9. The candidate set reduces to a failure of the back-left motor, increased friction in the back-left motor, and a bias or drift in the v_{BL} sensor. At 50.25 s, an increase in both i and i_{BL} are detected, eliminating the motor failure fault (which would have decreased the current) and the faults in the v_{BL} sensor (since they cannot affect any other sensors), leaving the motor friction fault as the sole candidate. The friction value is computed using the steady-state wheel speed equation, averaged from the time of detection to the present time. At 50.25 the friction value is estimated at 1.47 times its nominal value. By 100 s, the estimate is at 10.1 times its nominal value. HyDE is also able to diagnose this fault at 50.2. However, HyDE does not support fault magnitude estimation (identification). See Figure 12 for a diagram that illustrates the timing of various events in this scenario.

Prognosis begins once the fault is identified. Here, multiple possible future input trajectories may be assumed that will help with decision making. The increase in current due to the fault will cause the batteries to discharge earlier

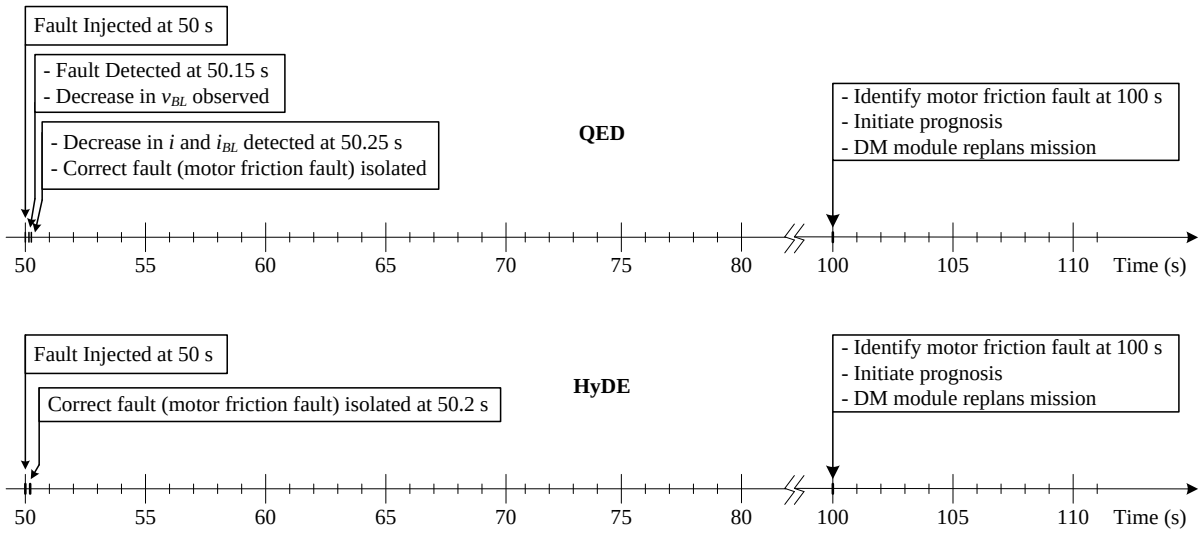


Figure 12. Timing diagram for the motor friction fault

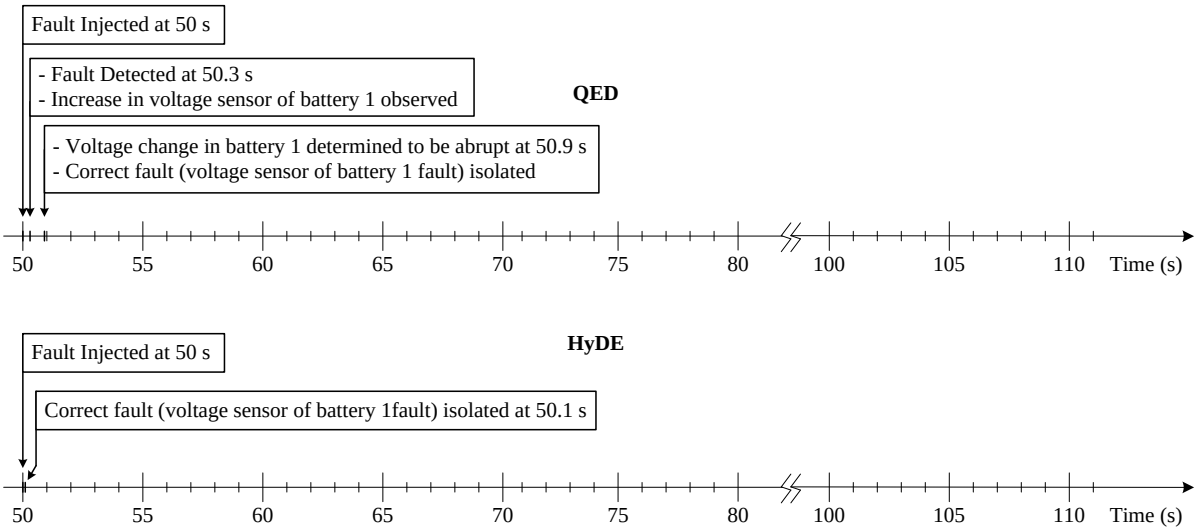


Figure 13. Timing diagram for the voltage sensor fault

than expected, and the increase in motor temperature may bring the motor to its EOL due to overheating. According to the prognosis algorithm, the overheating event occurs in 1423 s if the rover continues to travel at the same speed and at 4084 s at 80% speed (0.4 m/s). At 60% speed (0.3 m/s), the steady-state value of the temperature is below motor temperature limit, so the rover can travel indefinitely without overheating the motor, and EOL is determined solely by battery end-of-discharge, which ends up being 6313 s. However, if the friction-robust controller is used, then the overall current draw decreases and motor overheating can be prevented. Furthermore, less overall current is drawn from the batteries for the same forward speed of the rover, resulting in improved RULs of 4283 s at normal speed (0.5 m/s), 6004 s at 0.4 m/s, and 8897 s at 0.3 m/s.

If the non-robust LC is used, the decision-making module considers traverses with different speeds (0.3, 0.4, and 0.5 m/s) in order to arrive at the optimal payoff. Traveling at 0.5 m/s does not allow the rover to visit any of the other waypoints due to high energy draw. Traveling at 0.4 m/s gets the rover to waypoints 3 and 6, with $R(q_s)$=11. Reducing the speed further to 0.3 m/s allows the rover to get to a higher-reward (but more difficult to get to) waypoint 7 instead ($R(q_s)$=14). Consequently, the rover speed is reduced to 0.3 m/s. Introducing the friction-robust LC allows to keep the motor temperature within safe limits at any of the speeds considered and, due to the reduced power draw, makes it possible to accomplish longer traverses: $q_s = \{1, 2, 5\}$ at 0.5 m/s ($R(q_s)$=15); $q_s = \{1, 6, 3, 5\}$ at 0.4 m/s ($R(q_s)$=17); and $q_s = \{1, 3, 5, 2\}$ at 0.3 m/s ($R(q_s)$=18). Since travel time

is not one of the optimization constraints, the lower speed of 0.3 m/s is therefore selected (Figure 11).

8.3. Voltage Sensor Fault

As a third scenario, we consider a bias fault in the voltage sensor of battery cell 1. The bias has a value of 0.5 V and is injected at 50 s. QED detects the fault at 50.3 s, and, since no other faults can produce an increase in voltage, only bias and drift faults in the voltage sensor are valid candidates. At 50.9 s, the change is determined to be abrupt, isolating the bias fault as the only candidate. The bias value is estimated to be 0.51 V. HyDE is able to detect a voltage sensor fault on Battery 1 at 50.1. However the HyDE model does not try to further isolate the fault as a bias or drift. The timing of the events is shown in Figure 13.

Because the fault was determined to be in a sensor (and one that is not part of a control loop), the rover was deemed to be capable of operating without reconfiguration or mission changes required. Therefore, neither the prognoser nor the decision-making module were invoked.

In all of the above scenarios the $\mathcal{PC}$-based decision-making algorithm produced the same results as a (much slower) exhaustive search algorithm. The latter was used as a deterministic way to verify the general correctness of the former. The maximum number of the remaining waypoints was limited to 10 as beyond that running the exhaustive search algorithm became impractical (with execution times of 300 seconds or longer). Execution times for the $\mathcal{PC}$-based algorithm largely depended on the hyperparameter values used (see Balaban and Alonso (2012)) for a more in-depth discussion on the subject), however the algorithm appeared to be suitable for real-time use even on scenarios involving 25 waypoints (the maximum number attempted). While the results produced by $\mathcal{PC}$ and other approximate algorithms will degrade in accuracy and repeatability as the number of waypoints grows, the size of the search space in problems of this type may still leave them as the best option.

9. CONCLUSIONS

The work described in this paper is aimed at providing an inexpensive, safe platform for development, evaluation, and comparison of prognostics-enabled decision-making algorithms. Technologies resulting from this research are planned to be transferred for further maturation on unmanned aerial vehicles and other complex systems. At present, the K11 testbed already constitutes a promising platform for PDM research. A list of fault modes of interest has been identified and a number of them have been implemented in software and/or hardware. A software simulator has been developed that incorporates models of both nominal and off-nominal behavior, with the models validated using experimental data. The software architecture for the testbed has been defined in such a way as to allow quick replacement of autonomy elements depending on the testing objectives and customers. The sensor suite and the data acquisition system support a wide range of reasoning algorithms. The first set of such algorithms, for performing diagnostics, prognostics, and decision-making, is being deployed and tested. At this point it has been tested in simulation on scenarios involving battery parasitic load, increased motor friction, and voltage sensor faults.

Plans for the near future include addition of further hardware-injectable fault modes, field experiments of greater complexity, simulator model refinement, and extension of PDM methods to handle more complex problems, including constraints relaxation when the situation requires that. Data collected on the testbed is planned for distribution to other researchers in the field.

10. ACKNOWLEDGMENTS

The authors would like to gratefully acknowledge the contributions of researchers and student interns at NASA Ames Research Center: Bhaskar Saha, Sankalita Saha, Kai Goebel, Brian Bole, Terry Fong, Liam Pedersen, Vinh To, Susan Lee, Hans Utz, Lorenso Fluckiger, Maria Bulaat, Vytas SunSpiral, Jeremy Frank, Michael Dalal, Zachary Ballard, Sterling Clarke, Tabitha Smith, Kevin Rooney, and Sebastian Hening. Many ideas in this work have been borne out of discussions with colleagues at Impact Technologies (Liang Tang, Eric Hettler, Bin Zhang, and Jonathan DeCastro). The funding for this research is provided by NASA ARMD System-wide Safety & Assurance Technology (SSAT) project.

REFERENCES

Balaban, E., & Alonso, J. (2012, September). An Approach to Prognostic Decision Making in the Aerospace Domain. In *Annual Conference of the Prognostics and Health Management Society 2012.* Minneapolis, MN.

Balaban, E., Narasimhan, S., Daigle, M., Celaya, J., Roychoudhury, I., Saha, B., ... Goebel, K. (2011, September). A Mobile Robot Testbed for Prognostics-Enabled Autonomous Decision Making. In *Annual Conference of the Prognostics and Health Management Society 2011.* Montreal, Canada.

Balaban, E., Saxena, A., Bansal, P., Goebel, K., & Curran, S. (2009). Modeling, Detection, and Disambiguation of Sensor Faults for Aerospace Applications. *IEEE Sensors Journal*, *9*(12), 1907–1917.

Balaban, E., Saxena, A., Narasimhan, S., Roychoudhury, I., & Goebel, K. (2011). Experimental Validation of a Prognostic Health Management System for Electro-Mechanical Actuators. In *AIAA Infotech@Aerospace.*

Balaban, E., Saxena, A., Narasimhan, S., Roychoudhury, I., Goebel, K., & Koopmans, M. (2010, September). Airborne Electro-Mechanical Actuator Test Stand for De-

velopment of Prognostic Health Management Systems. In *Annual Conference of the Prognostics and Health Management Society 2010.* Portland, OR.

Bole, B., Tang, L., Goebel, K., & Vachtsevanos, G. (2011). Adaptive Load-Allocation for Prognosis-Based Risk Management. In *Annual Conference of the Prognostics and Health Management Society 2011.* Montreal, Canada.

Brown, D. W., Georgoulas, G., & Bole, B. (2009). Prognostics Enhanced Reconfigurable Control of Electro-Mechanical Actuators. In *Annual Conference of the Prognostics and Health Management Society 2009.* Denver, CO.

Brown, D. W., & Vachtsevanos, G. J. (2011). A Prognostic Health Management Based Framework for Fault-Tolerant Control. In *Annual Conference of the Prognostics and Health Management Society 2011.* Montreal, Canada.

Cassandra, A. R., Kaelbling, L. P., & Littman, M. L. (1995). Acting Optimally in Partially Observable Stochastic Domains. In *Proceedings of the National Conference on Artificial Intelligence.*

Celaya, J., Kulkarni, C., Biswas, G., & K., G. (2012). Towards A Model-based Prognostics Methodology for Electrolytic Capacitors: A Case Study Based on Electrical Overstress Accelerated Aging. *International Journal of the Prognostics and Health Management Society*, *3*(2-004).

Ceraolo, M. (2000, November). New Dynamical Models of Lead-Acid Batteries. *IEEE Transactions on Power Systems*, *15*(4), 1184–1190.

Chen, M., & Rincon-Mora, G. (2006, June). Accurate Electrical Battery Model Capable of Predicting Runtime and I-V Performance. *IEEE Transactions on Energy Conversion*, *21*(2), 504 - 511.

Daigle, M., & Goebel, K. (2011, March). Multiple Damage Progression Paths in Model-Based Prognostics. In *IEEE Aerospace Conference.*

Daigle, M., & Roychoudhury, I. (2010, October). Qualitative Event-based Diagnosis: Case Study on the Second International Diagnostic Competition. In *Proc. of the 21st International Workshop on Principles of Diagnosis* (pp. 371–378).

Daigle, M., Saha, B., & Goebel, K. (2012, March). A Comparison Of Filter-Based Approaches For Model-Based Prognostics. In *IEEE Aerospace Conference.*

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).

Edwards, D., Orchard, M. E., Tang, L., Goebel, K., & Vachtsevanos, G. (2010). Impact of Input Uncertainty on Failure Prognostic Algorithms : Extending the Remaining Useful Life of Nonlinear Systems. In *Annual Conference of the Prognostics and Health Management Society 2010.* Portland, OR.

Fudenberg, D., & Tirole, J. (1991). *Game Theory*. MIT Press.

Genta, G. (1997). *Motor Vehicle Dynamics: Modeling and Simulation*. World Scientific Publishing Company Inc.

Henning, M. (2004). A New Approach to Object-Oriented Middleware. *IEEE Internet Computing*, *8*(1), 66–75.

Hogge, E., Quach, C., Vazquez, S., & Hill, B. (2011). *A Data System for a Rapid Evaluation Class of Subscale Aerial Vehicle* (TM No. 2011-217145). NASA.

Huggins, R. (2008). *Advanced Batteries: Materials Science Aspects*. Springer.

Lachat, D., Krebs, A., Thueer, T., & Siegwart, R. (2006). Antarctica Rover Design And Optimization For Limited Power Consumption. In *4th IFAC Symp. on Mechatronic Systems.*

Mandow, A., Martínez, J., Morales, J., Blanco, J., García-Cerezo, A., & Gonzalez, J. (2007). Experimental Kinematics for Wheeled Skid-Steer Mobile Robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems* (p. 1222-1227).

Mosterman, P. J., & Biswas, G. (1999). Diagnosis of Continuous Valued Systems in Transient Operating Regions. *IEEE Trans. on Sys., Man and Cybernetics, Part A*, *29*(6), 554-565.

Narasimhan, S., & Browston, L. (2007). Hyde - A General Framework For Stochastic And Hybrid Model-based Diagnosis. In *18th International Workshop on Principles of Diagnosis* (p. 162-169).

Orchard, M. E. (2007). *A Particle Filtering-Based Framework for On-Line Fault Diagnosis and Failure Prognosis*. PhD Thesis, Georgia Institute of Technology.

Poll, S., et al. (2007, May). Evaluation, Selection, and Application of Model-Based Diagnosis Tools and Approaches. In *AIAA Infotech@Aerospace 2007 Conference and Exhibit.*

Rasmussen, C., & Williams, C. (2006). *Gaussian Processes for Machine Learning* (Vol. 1). MIT press Cambridge, MA.

Reveley, M. S., Kurtoglu, T., Leone, K. M., Briggs, J. L., & Withrow, C. A. (2010, December). *Assessment of the State of the Art of Integrated Vehicle Health Management Technologies as Applicable to Damage Conditions* (TM No. 2010-216911). NASA.

Saha, B., & Goebel, K. (2009). Modeling Li-Ion Battery Capacity Depletion in a Particle Filtering Framework. In *Annual Conference of the Prognostics and Health Management Society 2009.*

Saha, B., Quach, C. C., & Goebel, K. (2012, March). Optimizing Battery Life for Electric UAV's Using a Bayesian Framework. In *IEEE Aerospace Conference.*

Schwabacher, M. (2005). A Survey of Data-Driven Prognos-

tics. In *Proceedings of the AIAA Infotech@Aerospace Conference.*

Smith, M., Byington, C., Watson, M., Bharadwaj, S., Swerdon, G., Goebel, K., & Balaban, E. (2009). Experimental and Analytical Development of Health Management for Electro-Mechanical Actuators. In *IEEE Aerospace conference* (pp. 1–14).

Tang, L., Hettler, E., Zhang, B., & Decastro, J. (2011). A Testbed for Real-Time Autonomous Vehicle PHM and Contingency Management Applications. In *Annual Conference of the Prognostics and Health Management Society 2011.*

Tang, L., Kacprzynski, G. J., Goebel, K., Reimann, J., Orchard, M. E., Saxena, A., & Saha, B. (2007). Prognostics in the Control Loop. In *Working Notes of 2007 Fall AAAI Symposium* (pp. 128–135).

Wolpert, D. (2006). Information Theory–The Bridge Connecting Bounded Rational Game Theory and Statistical Physics. *Complex Engineered Systems*, 262–290.

BIOGRAPHIES

Edward Balaban is a research engineer in the Diagnostics and Prognostics group at NASA Ames Research Center. He received the Bachelor degree in Computer Science from The George Washington University in 1996 and the Master degree in Electrical Engineering from Cornell University in 1997. His main areas of interest are diagnostics, prognostics, and prognostics-enabled decision making. During his years at Ames he has participated in the research and development of system health management elements for the X-34 experimental reusable launch vehicle, the International Space Station, robotic astronaut assistants, autonomous planetary drills, and other aerospace applications. He is a member of the PHM Society, the IEEE, and the AIAA.

Sriram Narasimhan is a Project Scientist with University of California, Santa Cruz working as a contractor at NASA Ames Research Center in the Discovery and Systems Health area. He received his M.S and Ph.D. in Electrical Engineering and Computer Science from Vanderbilt University. He also has a M.S in Economics from Birla Institute of Technology and Science. His research interests are in model-based diagnosis with a focus on hybrid and stochastic systems. He is the technical lead for the Hybrid Diagnosis Engine (HyDE) project.

Matthew J. 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. 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.

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. Dr. Roychoudhury is a member of the Prognostics and Health Management Society and the IEEE. His research interests include hybrid systems modeling, model-based diagnostics and prognostics, distributed diagnostics and prognostics, and Bayesian diagnostics of complex physical systems.

Adam Sweet is a research engineer in the Diagnostics and Prognostics group at NASA Ames Research Center. He graduated with an MS in Mechanical Engineering from UC Berkeley in 1999, and has worked at Ames ever since. His project experience encompasses robotics, hybrid system simulation, model-based diagnosis, and flight software development for nanosatellites.

Christopher Bond is an electrical engineer with the Control and Data Systems group at NASA Kennedy Space Center. He received the B.S.E.E (summa cum laude) from the Georgia Institute of Technology in 1995. He has been involved with various projects throughout his career at NASA, including design modifications to and testing of the Space Shuttle ground landing systems, instrumentation for weather systems, as well as his current role as senior network engineer for the new Spaceport Command and Control system being built at Kennedy Space Center.

José R. Celaya is a research scientist with SGT Inc. at the Prognostics Center of Excellence, NASA Ames Research Center. He received a Ph.D. degree in Decision Sciences and Engineering Systems in 2008, a M. E. degree in Operations Research and Statistics in 2008, a M. S. degree in Electrical Engineering in 2003, all from Rensselaer Polytechnic Institute, Troy New York; and a B. S. in Cybernetics Engineering in 2001 from CETYS University, México.

George Gorospe received the B.E. degree in Mechanical Engineering from the University of New Mexico, Albuquerque, New Mexico, USA, in 2012. Since October 2012, he has been with the Universities Space Research Association, at NASA Ames Research Center. His current research interests include mission design, systems engineering, and autonomous mobile robot control.