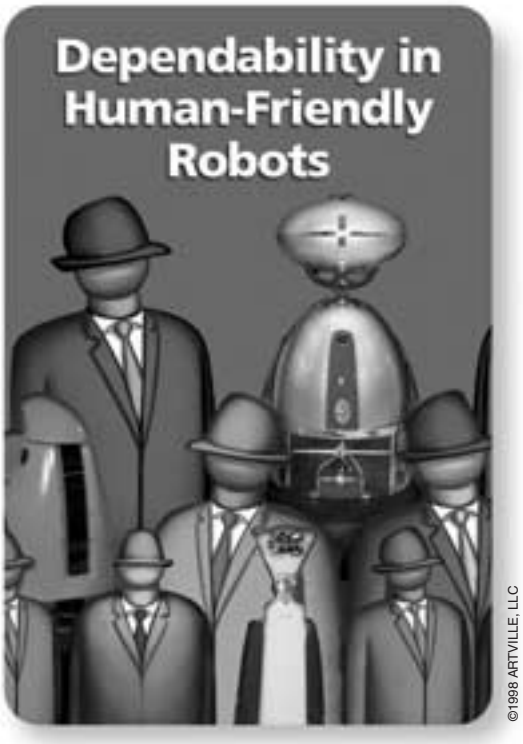
Tractable Particle Filters for Robot Fault Diagnosis

Real-Time Fault Diagnosis

By VANDI VERMA, GEOFF GORDON,
REID SIMMONS, and SEBASTIAN THRUN



This article presents a number of complementary algorithms for detecting faults on-board operating robots, where we define a fault as a deviation from expected behavior. Experience has shown that even carefully designed and tested robots may encounter faults [3]. One of the reasons for this is that components degrade over time. Another is that the operators of the robot rarely have complete knowledge of the environment in which it operates and, hence, may not have accounted for certain situations.

In a number of application domains, such as planetary exploration, search and rescue, mine mapping, nuclear waste cleanup, and demining, robots operate in environments where human intervention is expensive, slow, unreliable, or impossible. It is therefore essential for robots to monitor their behavior so that faults may be addressed before they result in catastrophic failures. This monitoring needs to be efficient since there is limited computational power available on robots. Not only are robots venturing into areas inaccessible or dangerous for humans, they are also increasingly becoming a part of day-to-day life. It is also important for these robots to detect faults in a timely manner, since failure to do so may result in expensive consequences, both monetary and in terms of consumer trust that may be hard to regain. If faults go undetected, autonomous robots in real-world environments may behave in an unpredictable or dangerous manner. On the other hand, detecting and recovering from faults can considerably improve the performance of the robots [2].

Fault detection and identification (FDI) for robots is a complex problem, because the space of possible faults is very large; robot sensors, actuators, and environment models are uncertain; and there is limited computation time and power. Probability theory provides a natural representation of the uncertainty in the rover domain, but an exact Bayesian solution to FDI is intractable. Traditional methods address this intractability by *approximating the problem* using linear approximations of rover dynamics and/or by ignoring uncertainty. Often, these approximations are unrealistic, and either faults go undetected, or an unreasonable number of false positives are triggered.

We instead prefer to *approximate the solution* using Monte Carlo methods. The rationale behind this is that the two stages of this problem, building models and online monitoring, have very different computational constraints. Theoretically, there is little restriction on the time and computation available for building models, since this may be done offline, but monitoring these models must be done in real time with the limited computation available on a robot. We consider approximate

 1070-9932/04/$20.00©2004 IEEE

inference using complex models a better trade-off than exact inference with simple models. Classical Monte Carlo methods for dynamic systems, such as particle filters, are capable of tracking complex nonlinear systems with noisy measurements. The problem is that estimates from a particle filter tend to have a high variance for small sample sets. Using large sample sets is computationally expensive and defeats the purpose.

We present a number of complementary algorithms for improving the accuracy of FDI with a computationally tractable set of samples in a particle filter. Experimental results in the rover domain show significant improvement in accuracy over the classical approach.

The algorithms described in this article enable detection of a wider range and larger number of faults during robot operation than has hitherto been possible. Furthermore, these algorithms provide the probability of the robot being in each of the fault and operational states given the sensor data. They can handle noisy sensors; nonlinear, non-Gaussian models of behavior; and are computationally efficient.

Estimating faults in terms of a probability distribution over all fault states captures the uncertainty in fault identification that results from noisy and insufficient data. In addition, it allows a lot of flexibility in the types of planners/controllers that may be used for controlling the robot and recovering from faults. For example, such distributions are compatible with classical conditional planners, or Markov decision processes (MDPs), which may use the most likely state to determine which action to take, and also partially observable MDPs (POMDPs) [67], which use the distribution over the entire state space.

Particle Filters for Monitoring Faults

Our formulation of the fault detection problem requires estimating robot and environmental states, as they change over time, from a sequence of sensor measurements that provide noisy, partial information about the states. The Bayesian approach to dynamic state estimation addresses this problem. Particle filters have been extensively used for Bayesian state estimation in nonlinear systems with noisy measurements [35], [22], [15]. They approximate the probability distribution with a set of samples or particles. The algorithms presented here all use particle filters.

Particle filters have a number of characteristics that make them attractive for fault detection on robots: they are nonparametric (can represent arbitrary distributions), can handle hybrid state spaces, can handle noisy sensing and motion, and can easily be extended to an anytime approach where the number of particles (hence, estimation accuracy) can be adjusted to match available computation.

FDI for Robots

A fault is defined as a deviation from the expected behavior of the system. Fault detection is defined as the process of determining that a fault has occurred. Fault identification is the process of determining exactly which exception or fault occurred [36]. FDI is typically passive, i.e., it does not alter control actions.

This article concentrates on a class of faults that are relatively difficult to detect because they cannot be inferred from sensor values at a given instance but require a sequence of time-varying sensor values. In addition, the expected behavior of the robot may be different in different operating conditions. For example, high power draw on flat ground may be cause for concern, but high power draw on a slope might be perfectly acceptable.

The faults addressed here include mechanical component failures, such as broken motors and gears; faults due to environmental interactions, such as a wheel stuck against a rock; and sensor failures, such as broken encoders.

FDI as Recursive State Estimation

State estimation is the process of determining the state of a system from a sequence of data. FDI has a natural representation as a state-estimation problem. We represent the possible fault and operational modes of the systems as explicit states. The sequence of measurements is then used to determine the state of the system. There are two main classes of state-estimation methods: batch- and recursive-estimation methods.

Batch methods treat all data with equal importance and find an optimal estimate of the state given the entire sequence. However, full batch estimation is computationally expensive and gets slower as the robot accumulates increasing volumes of data. It is, therefore, not suitable for FDI.

Recursive state-estimation methods make a Markov assumption, i.e., the past and future are conditionally independent given the current state. These methods incorporate the data as it becomes available and replace the data with a statistic. Estimates at subsequent timesteps use this statistic instead of the history of data for state estimation.

Classical Particle Filter

For FDI, we concentrate on a discrete-time, first-order Markov formulation of the state-estimation problem. The state being estimated is hybrid, i.e., it consists of both discrete and continuous components. Let $D = \{d_1 \ldots d_K\}$ represent K hidden discrete fault and operational states of the robot, $d_t \in D$ the discrete state of the robot at time t, and $d^t \in \{d_1, \ldots, d_t\}$ the discrete, first-order Markov chain representing the evolution of the state over time. In addition to the discrete states, there are also continuous states that track the dynamic behavior of the robot. Let $x_t \in \Re^{n_x}$ denote the multivariate continuous state at time t. The state of the robot is observed through a sequence of measurements, $z^t = \{z_1 \ldots z_t\}$, $z_t \in \Re^{n_z}$. Probabilistic models of the change in state over time (state transition model) and the relationship between the measurements and the state (measurement model) capture the inherent noise. State transitions depend on prior state, observation, and, in some cases, the control action. These models are assumed to be stationary, i.e., the models do not change with time. Bayesian filtering, represented by (1), provides a recursive estimate of the posterior (newly updated) probability distribution over the states. Here, control is omitted in equations for brevity. We consider the following factored representation, where the discrete state transition is conditionally independent of the continuous state transitions given the previous discrete state (This is not an exact factorization, but is reasonable in most cases.):

$$p(x_t, d_t \,|\, z_{1\ldots t}) = \eta_t \, p(z_t \,|\, x_t, d_t) \int \sum_{d_{t-1}} p(x_t \,|\, x_{t-1}, d_t) \, p(d_t \,|\, d_{t-1}) \, dx_{t-1}, \quad (1)$$

where η_t is a normalizing constant.

The discrete state of the robot determines the dynamic behavior of the robot. The continuous state transitions, denoted by $p(x_t \,|\, x_{t-1}, d_t)$, that model this dynamic behavior are, therefore, conditioned on the discrete state transitions. A sequence of sensor measurements provides information about the continuous state based on the conditional measurement model $p(z_t \,|\, x_t, d_t)$. The fault-detection problem is to provide a distribution over the discrete state set D at each time step. This is a marginal distribution of the posterior distribution in (1). There is no closed form solution to this equation, hence, we use a particle filter approximation. A particle filter is a Monte Carlo approximation of the posterior distribution in a Bayes filter. Particle filters approximate the posterior with a set of N fully instantiated state samples, or particles, $\{(d_t^1, x_t^1) \ldots (d_t^N, x_t^N)\}$ and importance weights $\{w_t^{[i]}\}$:

$$\hat{p}_N(x_t, d_t \,|\, z_{1\ldots t}) = \sum_{i=1}^{N} w_t^{[i]} \, \partial_{x_t^{[i]}, d_t^{[i]}}(x_t, d_t), \quad (2)$$

where $\partial(\bullet)$ denotes the Dirac delta function. It can be shown that as $N \to \infty$, the approximation in (2) approaches the true posterior density [70].

In order to create the updated particle set, particles must be drawn from the posterior distribution. However, it is difficult to draw particles from the true posterior. Instead we draw them from a more tractable approximation to this distribution distribution $q(\bullet)$, called the proposal (or importance) distribution. The importance weights are used to account for the discrepancy between the proposal distribution $q(\bullet)$ and the true distribution $p(x_t, d_t \,|\, z_{1\ldots t})$, and for a given particle $x_t^{[i]}, d_t^{[i]}$ the importance weight is:

$$w_t^{[i]} = \frac{p\left(x_t^{[i]}, d_t^{[i]} \,\middle|\, x_{t-1}^{[i]}, d_{t-1}^{[i]}, z_t\right)}{q\left(x_t^{[i]}, d_t^{[i]}\right)}. \quad (3)$$

The simplest and most widely used choice for the proposal distribution is the transition probability [1] [27] [40], which makes the importance weight equal to the likelihood of the observation. This formulation is used in the classical particle filter (CPF).

Draw initial set of particles $B_0 = \left\{d_0^{[i]}, x_0^{[i]}\right\}_{i=1\ldots N}$, from the prior distribution. The particle set B_t is then recursively drawn from B_{t-1} as follows:

1) Draw N particles $\{d_t^{[i]}, x_t^{[i]}\}_{i=1\ldots N}$ based on the previous posterior $B_{t-1} = \{d_{t-1}^{[i]}, x_{t-1}^{[i]}\}_{i=1\ldots N}$, as follows:

$$d_t^{[i]} \sim p\left(d_t \,\middle|\, d_{t-1}^{[i]}\right), \quad (4)$$

$$x_t^{[i]} \sim p\left(x_t \,\middle|\, x_{t-1}^{[i]}, d_t^{[i]}\right). \quad (5)$$

2) Assign an importance weight to each particle:

$$w_t^{[i]} \sim p\left(z_t \,\middle|\, x_t^{[i]}, d_t^{[i]}\right). \quad (6)$$

3) Resample, i.e., draw N particles from the weighted particle set with a probability proportional to the weights (6). Assign the particle set to B_t with $w_t^{[i]} = 1$.

As mentioned previously, particle filters have computational and representational advantages over other Bayesian techniques. The main problem is that a large number of particles are often needed to maintain a reasonable approximation of the state probability distribution and to enable detection of rare events. Reliably tracking multiple low-probability fault states, which requires maintaining and updating large numbers of particles, is typically not practical due to limited computation. Small particle sets do not provide reasonable approximations because: 1) they are unlikely to represent low probability fault states, and 2) their estimates are likely to have a high variance.

Below, we describe algorithms that address these issues. First, the risk-sensitive particle filter (RSPF) reduces the number of particles required to track unlikely but important states by including risk estimates when updating the particles. Second, the variable-resolution particle filter (VRPF) reduces the number of particles required by dynamically grouping multiple low-probability states based on similarity. Third, looking at the expected sensor measurement one step ahead in time can improve the state estimate provided by a limited number of particles.

Risk-Sensitive Particle Filter

RSPFs [75] [71] incorporate a model of cost when generating particles. This approach is motivated by the observation that the cost of not tracking hypotheses is related to risk. Not tracking a rare but risky state may have a high cost, whereas not tracking a rare but benign state may be irrelevant. Incorporating a cost model into particle filtering improves the tracking of states that are most critical to the performance of the robot.

Faults are low-probability, high-cost events. The CPF generates particles proportional only to the posterior probability of an event. Therefore, monitoring a system to detect and identify faults based on a CPF requires a very large number of particles and is computationally expensive. RSPFs generate particles by factoring in the cost. Since faults have a high cost, even though they have a low probability, the RSPF ensures that particles will be generated to represent them. Therefore, a smaller number of particles than the CPF may be used to monitor these events. The approximate nature of the particle representation may result in sub-optimal control and, hence, increased cost. The goal of risk-sensitive sampling is to generate particles that minimize the cumulative increase in cost due to the approximate particle representation.

In more detail, risk-sensitive sampling uses a positive risk function $r(d)$ to generate more particles in high-risk regions of the state space. Fault states will have high $r(d)$ values and so will get proportionally more particles. The normal operational

mode will have a smaller $r(d)$ value and, therefore, proportionally fewer particles. Instead of tracking the posterior distribution, the RSPF tracks a product of the posterior and the risk:

$$\gamma_t \; r(d_t) \; p\left(d_t, x_t \,\middle|\, z^t\right), \tag{7}$$

where, γ_t is a normalization constant that ensures that (7) is a probability distribution. The choice of risk function is important. Thrun et al. in [71] present a method for obtaining this risk function via an MDP that calculates the approximate future risk of decisions made in a particular state.

Variable Resolution Particle Filter

Consider the problem of diagnosing locomotion faults on a robot. Motors on any of the wheels may stall at any time, but the probability of a stalled motor at any time is low. Failures in wheels on the same side of the robot generate similar observations (drift toward the side with the stuck wheel). Under these conditions, a CPF with small number of particles is likely to produce an estimate with high variance and identify some arbitrary wheel fault on the same side rather than the correct fault.

The VRPF [76] introduces the notion of abstract particles, in which a particle may represent an individual state or a set of similar states. With this method, a single abstract particle simultaneously tracks multiple similar states. A limited number of particles are, therefore, sufficient for representing large portions of the state space when likelihood of occupying this part of the state space is low. When the likelihood of the grouped states increases, and it is important to determine the relative importance of these states, particles are specialized to represent individual states. We show that tracking multiple states by grouping them together may at times reduce the state-estimation error. This is because the variance of the state estimate can often be dramatically reduced by abstraction, with only a minor increase in bias. We provide a quantitative decomposition of the error in terms of bias and variance. A bias-variance trade-off is made to dynamically refine and abstract states to change the resolution to minimize the state-estimation error. As a result, reasonably low variance posterior estimates can be obtained with a relatively small number of particles.

The VRPF requires a variable resolution state space model. We use a multilayered hierarchy: each physical (nonabstract) state corresponds to a leaf of the hierarchy. Sets of states with similar state transition and observation models are aggregated together at each nonleaf node in the hierarchy to form abstract states (see Figure 1). In addition to the physical state set D, the variable resolution model consists of a set of M abstract states $A = \{a_{(1)} \ldots a_{(M)}\}$ that represent sets of states and/or other abstract states:

$$a_j = \begin{cases} d_{(k)} \\ \bigcup_i \; a_{(i)}. \end{cases} \tag{8}$$

A (somewhat simplified) algorithm for abstracting and refining states in the VRPF is as follows.

The initial set of particles $B_0 = \{a_0^{[i]}, x_0^{[i]}\}_{i=1\ldots N}$ are drawn from the prior distribution. R_0 is set to the set of unique states (physical or abstract) represented in B_0. The particle set B_t is then recursively drawn from B_{t-1} as follows:

1) Project all the particles to physical states to use the physical transition and measurement models. If a particle, $a_{t-1}^{[i]} = d_{(j)} \in B_{t-1}$, is in an abstract state, one of its descendant physical states $\{d_{(j)}\}$, is selected proportional to the prior probability of the physical states as follows:

$$p\left(d_{t-1}^{[i]}\right) \sim \frac{\pi(d_{(j)})}{\sum_{d_k} \pi(d_{(k)})}; \quad d_j, d_k \in a_{t-1}^{[i]}.$$

Here $\pi(d)$ represents the prior probability of state d.

2) Update, weigh, and resample the projected particles as in steps 1, 2, and 3 from the CFP algorithm [(4)–(6)].
3) Re-assign particles to the appropriate abstractions of these physical states to reflect the current resolution of the state space (represented by R_{t-1}).

$$\text{If } d_t^{[i]} \in R_{t-1}, \text{ set } a_t^{[i]} = d_t^{[i]},$$

else find $p = \text{parent}\,(d_t^{[i]})$, where $p \in R_{t-1}$ and set $a_t^{[i]} = p$.

4) Vary the resolution of the abstraction to trade bias against variance and obtain a new set of states R_t. Steps 1–3 estimate a probability distribution over the state at a fixed resolution, represented by R_{t-1}. If the loss for representing using the parent is lower than that for the current state, the state is abstracted to the parent state. If the loss for using a child is lower than that for the current state, the state is refined to the child states. Details may be found in [76].

Step 4 requires a computation of loss. The loss of an abstract state $a_{(j)}$ is computed as the weighted sum of the loss of the physical states. Omitting details, which may be found in [76], we calculate loss as the sum of bias and variance. The generalization to abstract states biases the distribution over the physical states to the prior distribution. In other words, the abstract state has no information about the difference in probability given the data, of the states that it represents in abstraction. Instead it uses the prior distribution to project its posterior into the physical layer. Depending on the deviation from the prior, tracking in abstraction may increase the loss

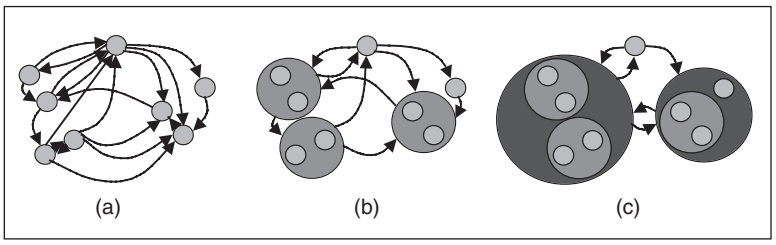


Figure 1. *(a) General Markov model representing physical states. (b) Abstract model with similar states grouped. Circles that enclose other circles represent abstract states. (c) Multiple levels of abstraction of the physical model in (a).*

from bias. But the variance of this estimate can be high, especially with small particle sizes. Hence we trade off bias and variance to minimize loss.

One-Step Look-Ahead Using an Unscented Kalman Filter

The approach described here is based on the observation that looking ahead at the measurements that result as a consequence of a fault can dramatically improve diagnosis. We describe a particle filter that generates particles at time t using a proposal distribution that takes into account the measurement at time t, in addition to the state at time $t-1$. Doucet et al. [16] show that this proposal distribution $p(x_t, d_t | x_{t-1}^{[i]}, d_{t-1}^{[i]}, z_t.)$ is the "optimal" proposal distribution, that is, the distribution that minimizes the variance of the importance weights conditioned on $x_{t-1}^{[i]}$, $d_{t-1}^{[i]}$ and z_t. Unfortunately, it is difficult to sample from this proposal distribution exactly, so we approximate it using unscented Kalman filters (UKFs) [37]. This approximation is similar to an unscented particle filter [74], but it takes into account the fact that some of our state variables are discrete while others are continuous. From [77] we have:

$$p\left(x_t, d_t \middle| x_{t-1}^{[i]}, d_{t-1}^{[i]}, z_t\right) = \eta^{[i]}\, \eta_{d_t}^{[i]}\, p\left(x_t \middle| x_{t-1}^{[i]}, d_t, z_t\right)\, p\left(d_t \middle| d_{t-1}^{[i]}\right), \tag{9}$$

where $\eta^{[i]}$ and $\eta_{d_t}^{[i]} = p(z_t | x_{t-1}^{[i]}, d_t.)$ are normalizing constants. We can ignore $\eta^{[i]}$ since it doesn't depend on x_t or d_t. The discrete transition probability $p(d_t | d_{t-1}^{[i]}.)$ is known, so, we will use UKFs to approximate $\eta_{d_t}^{[i]}$ and $p(x_t | x_{t-1}^{[i]}, d_t, z_t.)$ for each particle i and possible discrete transition d_t.

To compute these approximations, we need to examine each possible pair of i and d_t separately; this process can be computationally expensive, a complaint which we will return to later.

The UKF is a recursive minimum mean square error estimator that often provides an improvement over the extended Kalman filter (EKF) for nonlinear models. The EKF linearizes the nonlinear process and measurement models using the first-order terms of a Taylor series expansion. The UKF, on the other hand, does not approximate the nonlinear process and measurement models. It uses the actual models and instead approximates the distribution of the state variable as a Gaussian. The Gaussian approximation is specified using a minimal set of deterministically chosen samples called sigma points. Each sigma point is independently propagated through the process and measurement models, and the set of propagated sigma points is analyzed to provide a posterior Gaussian approximation.

In our case, we use the UKF to approximate $p(x_t | x_{t-1}^{[i]}, .d_t, z_t)$ as a Gaussian. This approximation will usually be excellent because we are conditioning on a single previous state and a single possible fault. Given this approximation, we can compute analytically the values for $\mu_{d_t}^{[i]}$ (the mean of x_t), $P_{d_t}^{[i]}$ (the covariance of x_t), and $\eta_{d_t}^{[i]}$ (the observation likelihood). Once we have finished our UKFs, we can sample from our proposal distribution by first drawing d_t and then x_t according to (10) and (11), then computing importance weights via (3).

$$\hat{p}\left(d_t \middle| d_{t-1}^{[i]}\right) \sim p\left(d_t \middle| d_{t-1}^{[i]}\right) \eta_{d_t}^{[i]} \tag{10}$$

$$\hat{p}\left(x_t \middle| x_{t-1}^{[i]}, d_t\right) \sim N\left(\mu_{d_t}^{[i]}, P_{d_t}^{[i]}\right). \tag{11}$$

In order to find $\eta_{d_t}^{[i]}$ for every d_t and i, we must compute a UKF for each particle and every possible next discrete state transition. Given that there is potentially a large number of faults to transition to at any step, this may be inefficient. We use the VRPF to address this problem. The VRPF reduces the number of next state transitions and, hence, UKF computations. This one-step look-ahead algorithm with VRPF is called VUF.

Results

Risk-Sensitive Particle Filter

In a simulation of the Hyperion robot [63], we explicitly introduced faults and recorded a sequence of controls and measurements that were then tracked by a RSPF and a CPF. In the experiment, the robot was driven with a variety of different control inputs in the normal operation mode. For this experiment, the measurements were the rover pose $(x, y, \square)$ and steering angle. At the 17th time step, Wheel 3 becomes stuck and locked against a rock. The wheel is then driven in the backward direction, fixing the problem. The robot returns to the normal operation mode and continues to operate normally until the gear on Wheel 4 breaks at the 30th time step. Figure 2(b) and (c) shows that RSPF yields superior results to CPF. Even though failures are very unlikely, the RSPF successfully identifies them due to the high risk associated with such failures; the CPF essentially fails to do so. The estimation error is shown in the bottom row of Figure 2(b) and (c). It is practically zero for the RSPF when 1,000 or more particles are used. CPF exhibits non-zero error even with 100,000 particles.

The second domain we use in our experiments is diagnosing locomotion faults in a physics-based simulation of a six-wheel robot. Figure 3(a) shows a snapshot of the robot in the Darwin2K [43] simulator. In simulation, we modeled stuck wheel faults at each wheel. Each of these faults causes a change in the robot dynamics, as shown in Figure 3(b). Note that faults on each side (right and left) have similar dynamics. The Markov model representing the discrete state transitions consists of seven physical states. As shown in Figure 3(c), the normal driving (ND) state may transition back to the normal driving state or to any one of six fault states: right front (RF), right middle (RM), right rear (RR), left front (LF), left middle (LM), and left rear (LR) wheel stuck. For this experiment, the measurements are changes in robot position and orientation.

In the first experiment in this domain, we represented (and induced) only two of the six faults in our state space: one on a wheel on the right and the second on a wheel on

the left. Figure 4(a) shows the improvement in performance with the RSPF.

When there are numerous similar fault states with high risk, the performance of the RSPF degrades. To demonstrate this we represented and tracked all six faults shown in Figure 3(b). The degradation in performance is shown in Figure 4(b) and (c). This degradation occurs because the RSPF works by trying to populate all of the fault states with particles; when there are too many fault states compared to the number of particles there is no way to populate all of them. The VRPF is designed to address this difficulty; the next section describes our experimental results with the VRPF.

Variable Resolution Particle Filter

Given that the three wheels on each side of the robot have similar dynamics, the VRPF uses a hierarchy that clusters the fault states on each side together. Figure 3(d) shows this hierarchical model, where the abstract states right side fault (RS), and left side fault (LS) represent sets of states {RF, RM, RR} and {LF, LM, LR} respectively. The highest level of abstraction therefore consists of nodes {ND, RS, LS}. Figure 3(e) shows how the state space in Figure 3(d) would be refined if the loss in abstract state RS given the number of particles is greater than the combined loss of the physical states RF, RM, and RR.

When particle filtering is performed with the variable resolution particle filter, the particles are initialized at the highest level in the abstraction hierarchy. When an RF fault occurs, this should result in a high likelihood of particles in RS. These particles should multiply, which should result in the bias in RS exceeding the reduction in variance in RS over RF, RM, and RR, thus favoring tracking at the finer resolution. Additional observations should then assign a high likelihood to RF.

Figure 5 shows a comparison of the error from monitoring the state using a CPF that tracks the full state space, and the VRPF that varies the resolution of the state space. The X axis shows the number of particles used; the Y axis shows the error in terms of KL divergence from an approximation of the true posterior computed using a large number of particles. One million particles were used to compute an approximation to the true distribution. The KL divergence is computed over the entire length of the data sequence and is averaged over multiple runs (varying from 50 to five as particle size was increased) over the same data set. The data set includes normal operation and each of the six faults. Figure 5(a) demonstrates that the performance of the VRPF is superior to that of the classical filter for small particle sizes. In addition Figure 5(b) shows the KL-divergence along the Y axis and wall clock time along the X axis.

One-Step Look-Ahead

The faults and the abstraction hierarchy considered in this experiment are the same as in the previous experiment. The measurements in this experiment were noisy estimates of the robot position and orientation. The particle set representing the state consists of N particles, where each particle $\lfloor a_t^{[i]}, x_t^{[i]} \rfloor$ is a hypothesis about the current state of the

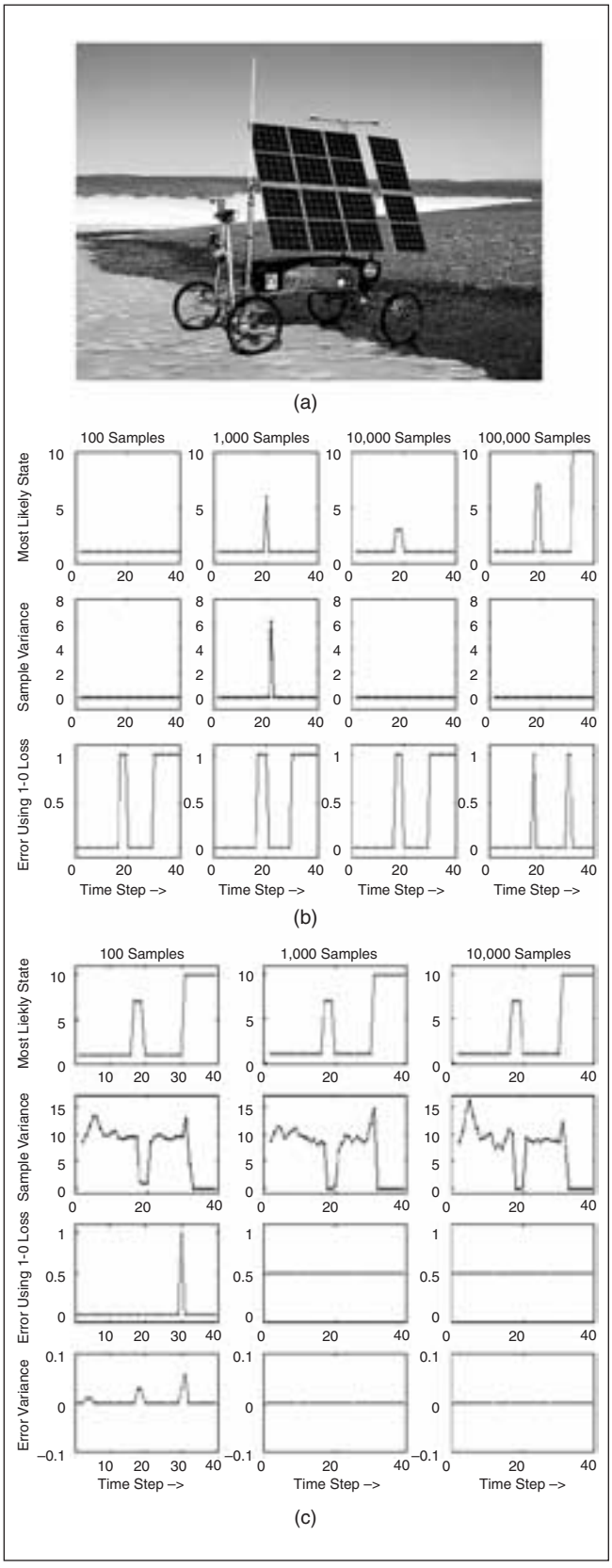


Figure 2. *(a) Hyperion robot. (b) Results with a simple particle filter. The state numbers represent different fault modes, with State 1 corresponding to normal operation. (c) Results with a RSPF.*

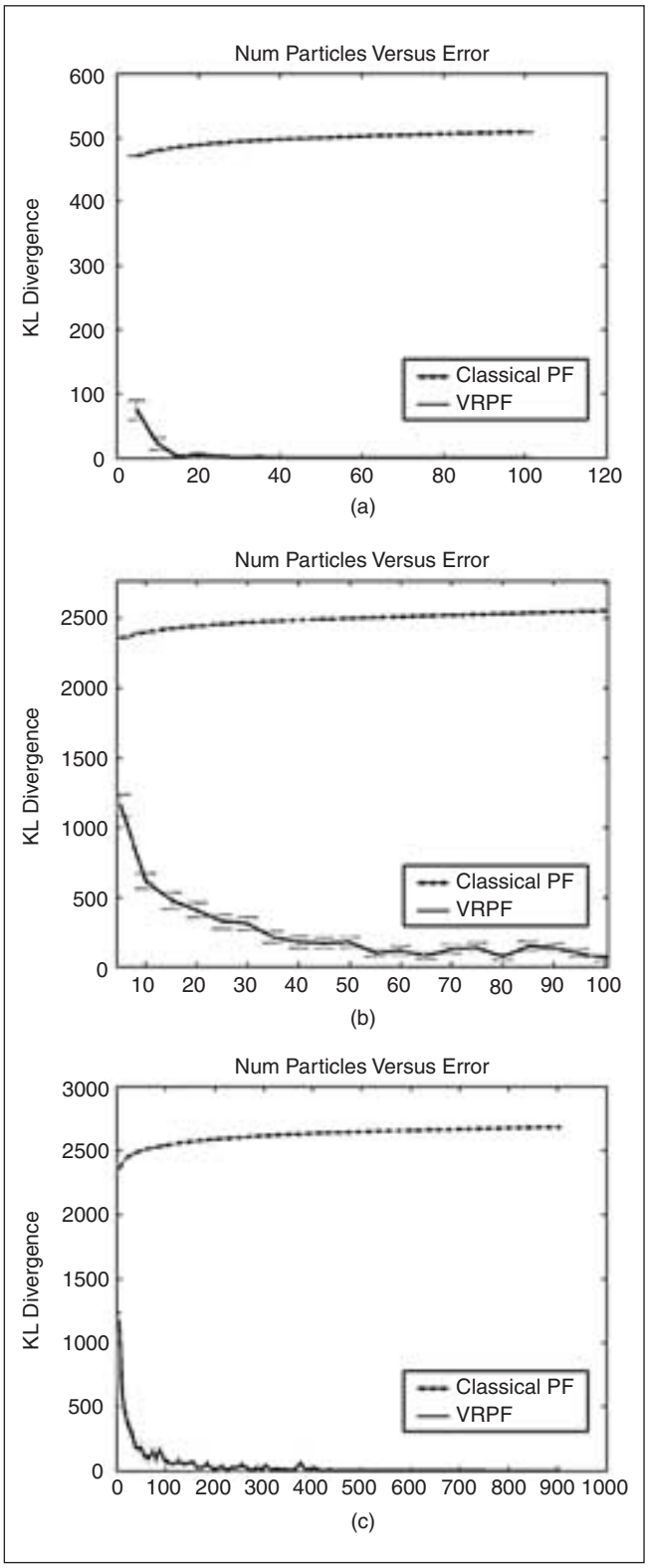


Figure 3. *Comparison of error in terms of KL-divergence between CPF and RSPF: (a) Improvement over CPF for small number of faults, (b) performance with numerous high risk fault states, (c) extended version of graph in (b). The CPF only tracked the normal state. The reason the KL-divergence increases slightly is because we use a Dirichlet 1 prior and the effect of the prior decreases with increasing sample sizes.*

system. $a_t^{[i]}$ is the discrete fault or normal state, and $x_t^{[i]}$ is the multidimensional continuous state representing the change in position and orientation of the robot.

Figure 6(a) and (b) shows a comparison of the performance of the CPF, UPF, and a VRPF with one-step look-ahead (VUF). The X axis shows the number of particles used. The Y axis shows the KL divergence from an approximation of the true posterior computed using a large number of particles. For the experiment in Figure 6(a), the continuous measurements were the absolute robot position. One million particles were used to compute an approximation to the true distribution. The KL divergence is computed over the entire length of the data sequence and is averaged over multiple runs over the same data set. The data set included normal operation and each of the six faults.

The improvement in performance of the VUF over UPF is expected to be even greater when the variable resolution state space model is larger and results in a larger reduction in the size of the state space than the simple experiment we present. This is because a UKF needs to be computed for every possible discrete state transition. The discrete state transitions between the abstract states in the VUF are smaller than the discrete state transitions between the physical states used in UPF.

Related Work

Other Particle Filter Based Approaches

Several other researchers have investigated the use of particle filters for robot fault diagnosis. Dearden and Clancy [13] address the issue of low probability faults by using an "oracle" to provide a set of candidate states that the system might end up in given the current distribution over the state space. On resampling, they ensure that there are always some particles in the states provided by the oracle. For sparse discrete state transitions, they do a one-step look-ahead to determine these states, but this can be expensive in large state spaces. A similar approach to diagnose a rocket propulsion system is used in [41].

N. de Freitas [10] presents an approach based on Rao-Blackwellised particle filters (RBPFs) [15]. RBPFs factor the state space, so that a subset of the variables may be computed analytically given the values of the rest of variables, which are obtained from particle filtering. The approach presented in [10] combines a particle filter to compute the distribution of the discrete states with a bank of Kalman filters to compute the distribution of the continuous states. This approach to include one-step look-ahead is extended in [49], which significantly improves performance. Funiak and Williams [23] combine this approach with Probabilistic Hybrid Automata [32] to allow continuous variables to affect discrete state transitions. All these approaches are restricted to linear, Gaussian models for each fault state. Hutter and Dearden [34] further extend this approach to moderately nonlinear models by using UKFs instead of Kalman filters. Each fault still has a Gaussian posterior.

Other Approaches to Fault Diagnosis

Most researchers in fault diagnosis have used techniques other than particle filtering. Some of these approaches are based on

very different assumptions and may not be entirely applicable to our domain, but provide valuable insights. Historically autonomous diagnosis has been a central theme in AI. Although these methods addressed very different problems, involving static environments (that do not evolve over time) and/or qualitative inputs.

Some methods classify based only on current sensor readings. Geometric classifiers divide the space of sensor measurements into different nominal and fault states [9]. These methods typically do not take temporal information into account, making them unsuitable for robot diagnosis.

Rule-based systems like MYCIN [65] and a number of expert systems [31] have been extensively used in diagnosis. These systems do not model uncertainty in the sensed values, although they may include hypotheses about sensor failure. Murphy and Hershberger in [56] present an approach that uses information from other sensors to test hypotheses about sensor failure.

Model-based reasoning, developed by Brown et al. in [64], inspired an entire field of research in diagnosis [11], [48], [24]. What defines model-based approaches is that they reason about explicit models of the system. Model-based diagnosis systems have been extended to handle temporal constraints in dynamic systems, e.g., XDE [30], SIDA [29], GMODS [33], SHERLOCK [12] and DIANA [8]. To address continuous dynamic systems, these systems use qualitative models, e.g., TEXSYS [25], MIMIC [18], and the Livingstone system [80]. State estimation in Livingstone is a search over the transitions of the hardware model to find a state that is consistent with the sensor measurements. The state constraints are formalized as logic formulae defined over the space of discretized variables representing the state of the system. Livingstone assumes that there is no uncertainty in the discretized sensor measurements and uses them as hard constraints. This results in an empty hypothesis space or incorrect identification if the true trajectory is not the most likely one, given only the current sensor measurements. Later enhancements made in Livingstone II [42] regenerate past hypotheses based on saved history. Livingstone was successfully tested on the Deep Space I spacecraft, but when tested on a rover, it triggered numerous false positives in the presence of noise [78].

Bayesian belief networks have also been used for diagnosis [68], [69]. Most applications of belief nets did not include temporal dependencies. Recently, dynamic Bayesian networks have been used to address temporal dependencies [44]. Other probabilistic approaches, such as [65] and [28], use Dempster-Shafer.

Kalman filters [38] are a popular method for tracking the state of a dynamic system and are optimal if the process and measurement equations are linear and the noise is Gaussian. There are various approaches that use a mixture of Kalman filters [60], [44], [10], for tracking multiple hypotheses about the state of the system. In some cases, where the number of faults and nominal states being tracked is small, all the filters may be run simultaneously. A probabilistic combination of the residual from the bank of Kalman filters can be used to determine the correct fault state, as in [60]. In a later extension, a backpropagation neural network was used to process these residuals and identify the fault [26]. But, in general, this is not very scalable. There is a family of algorithms that propagate the exact posterior for one step, giving a large Gaussian mixture, and then reduce

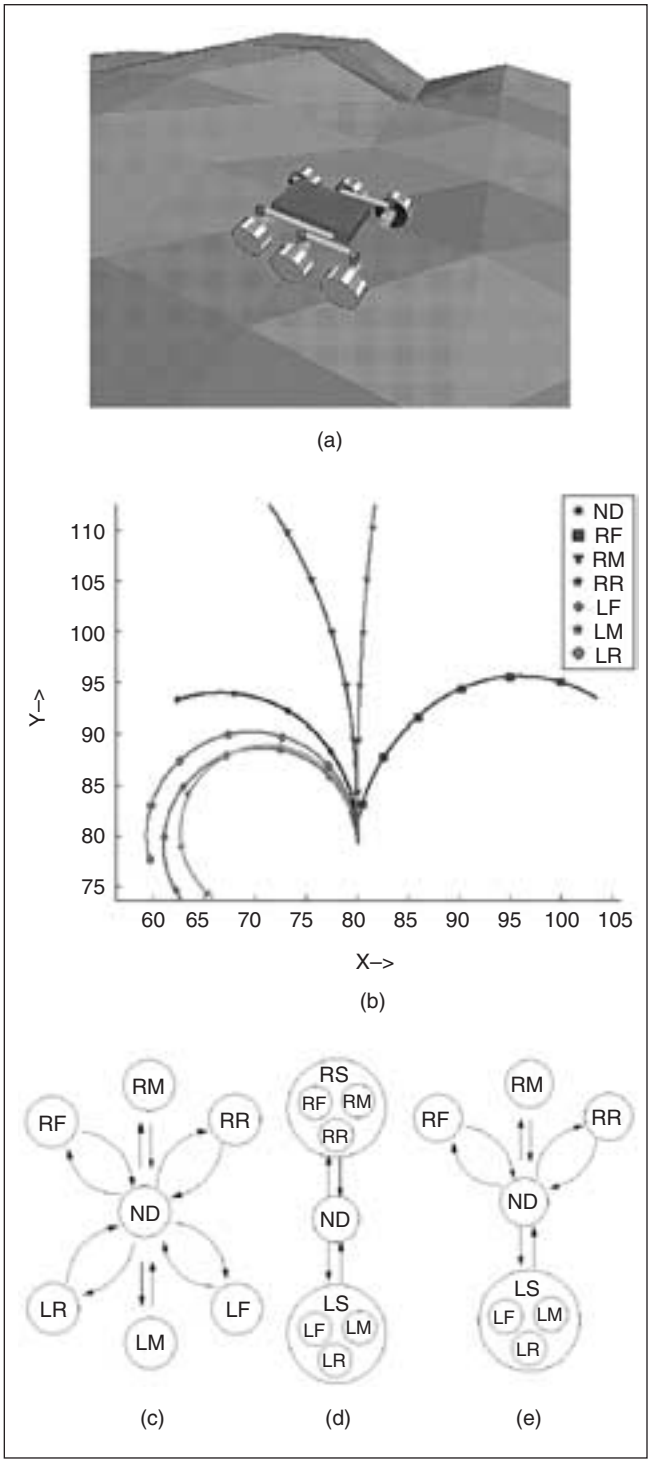


Figure 4. *(a) Robot in simulation. (b) Example of normal trajectory (ND) and the change in the same trajectory with a fault at each wheel. (c) Original discrete state transition model. (d) Abstract discrete state transition model. (e) State space model where RS has been refined. Self transitions have been excluded for clarity.*

the mixture as needed. Methods for reducing the mixture include: pruning to discard filters with very high residual [79], sampling mixture components according to weight [10], and repeatedly merging the most similar pair of mixture components [44]. Washington [79] uses a POMDP to represent the discrete state model; Haufbaur uses Probabilistic Hybrid Automata [32], which extends hidden Markov models (HMMs) with continuous dynamic models. The common shortcoming of all these methods is that they are typically restricted to linear models and Gaussian posterior for the dynamics. Another approach that addresses uncertainty and uses a POMDP representation is [21]. The focus of this approach was on recovery from faults and it operates on a discrete state space.

Conclusions

The algorithms presented focus on faults that cannot directly be detected from current sensor values but require inference from a sequence of time-varying sensor values. Each algorithm provides an independent improvement over the basic approach. These improvements are not mutually exclusive, and the algorithms may be combined to suit the application domain.

All of the approaches presented here require dynamic models representing the behavior of each of the fault and operational states. These models can be built from analytical models of the robot dynamics, data from simulation, or from the real robot. All the approaches presented detect faults from a finite number of known fault conditions, although there may potentially be a very large number of these faults.

Acknowledgments

We thank John Langford for collaborating on some aspects of this work; anonymous RAS reviewers and Ashley Stroupe for useful comments on this article; and Daniel Clancy, Richard Dearden, and Tom Minka for useful discussions.

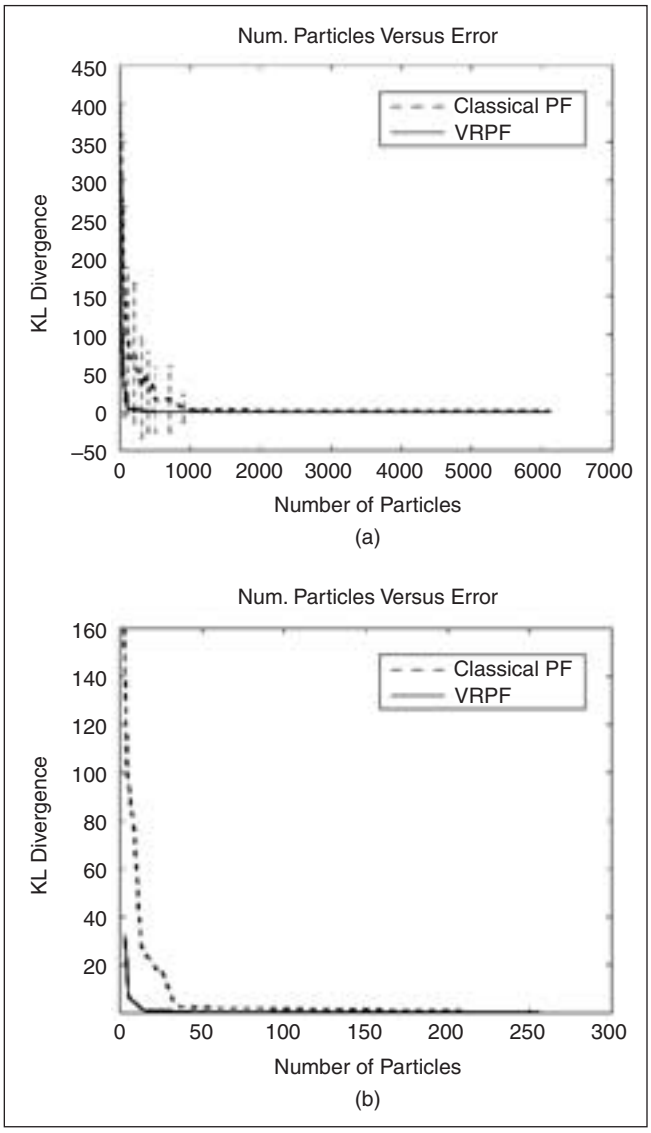


Figure 5. *Comparisons of the KL divergence from the true distribution: (a) CPF versus VRPF. number of particles used, (b) CPF versus VRPF, wall clock time.*

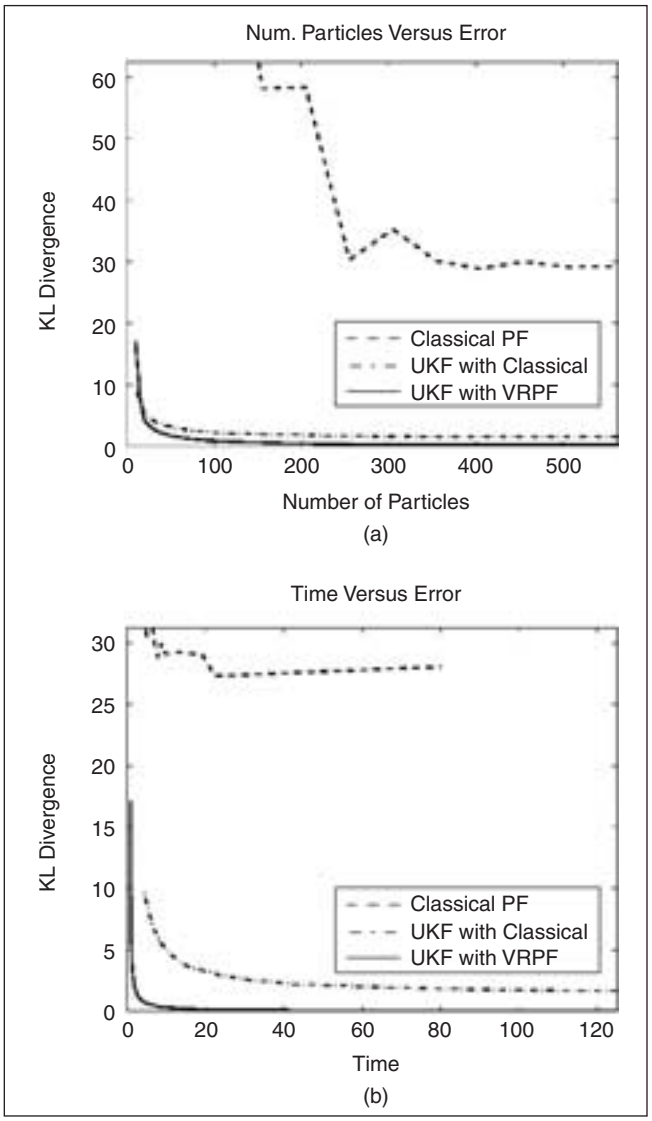


Figure 6. *Comparisons of the KL divergence from the true distribution: (a) UPF versus VUF, number of particles used, (b) UPF versus VUF, wall clock time.*

Keywords

Robot fault diagnosis, real-time fault detection, tracking hybrid dynamic systems, particle filters, efficient monitoring of hybrid dynamic systems.

References

[1] D. Avitzour, "A stochastic simulation Bayesian approach to multi-target tracking," *IEEE Trans. Aerosp. Electron. Syst.*, 1995.

[2] C. Bererton and P. Khosla, "An analysis of cooperative repair capabilities in a team of robots," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 2002.

[3] J. Carlson and R. Murphy, "Reliability analysis of mobile robots," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 2003.

[4] J. Carpenter, P. Clifford, and P. Fearnhead, "An improved particle filter for non-linear problems," *Proc. Inst. Electr. Eng. Proc.—Radar, Sonar, and Navigation*, 1999, no. 147, pp. 2–7.

[5] A.R. Cassandra, M. Littman, and N. Zhang, "Incremental pruning: A simple, fast, exact method for partially observable Markov decision processes," in *Proc. 13th Annual Conf. on Uncertainty in Artificial Intelligence (UAI-97).*

[6] A.R. Cassandra, "Exact and approximate algorithms for partially observable Markov decision process," Brown University, Providence, Rhode Island, May 1998.

[7] J. Chen and R.J. Patton, *Robust Model-Based Fault Diagnosis for Dynamic Systems*, Norwell, MA: Kluwer, 1998.

[8] P. Dague, O. Jehl, and P. Taillibert, An Interval Propagation and Conflict Recognition Engine for Diagnosing Continuous Dynamic Systems, Expert Systems in Engineering: Lecture Nites in AI, 1990.

[9] B. Dasarthy, *Nearest Neighbor Pattern Classification Techniques*. Los Alamitos, CA: IEEE Comput. Soc. Press, 1990.

[10] N. de Freitas, *Rao-Blackwellised Particle Filtering for Fault Diagnosis*, IEEE Aerospace, 2002.

[11] J. de Kleer and B.C. Williams, "Diagnosing multiple faults," Readings in Nonmonotonic Reasoning, 1987.

[12] J. de Kleer and B.C. Williams, "Diagnosis with behavioral modes," in *Proc. 11th Int. Joint Conf. on Artificial Intelligence*, 1989.

[13] R. Dearden and D. Clancy, "Particle filters for real-time fault detection in planetary rovers," *12th Int. Workshop on Principles of Diagnosis*, DX-2002.

[14] A. Doucet, N. de Freitas, and N. Gordon, *Sequential Monte Carlo Methods in Practice*. Berlin, Germany: Springer-Verlag, 2001.

[15] A. Doucet, N. de Freitas, K. Murphy, and S. Russel, *Rao-Blackwellised Particle Filtering for Dynamic Bayesian Networks*, UAI, 2000.

[16] A. Doucet, *On Sequential Simulation-based Methods for Bayesian Filtering*. Cambridge, U.K.: Cambridge Univ. Press, 1998.

[17] A. Drake, *Observation of a Markov Process Through a Noisy Channel*. Cambridge, MA: MIT Press, 1962.

[18] D. Dvorak and B.J. Kuipers, *Model-Based Monitoring of Dynamic Systems*, IJCAI, 1989.

[19] G.E. Monahan, "A survey of partially observable Markov decision processes: Theory, models and algorithms," *Manage. Sci.*, 1982.

[20] J.L. Feranández and R.G. Simmons, "Robust execution monitoring for navigation plans," in *Proc. 1998 IEEE/RSJ Int. Conf.,* 1998.

[21] J.L. Feranández, "Supervision, detection, diagnosis and exception recovery in autonomous mobile robots," University of Vigo, March 2000.

[22] D. Fox, W. Burgard, and S. Thrun, "Markov localization for mobile robots in dynamic environments," *J. Artif. Intell.*, vol. 11, pp. 391-427, 1999.

[23] S. Funiak and B.C. Williams, "Multi-modal particle filtering for hybrid systems with autonomous mode transitions," in *Proc. Workshop on Principles of Diagnosis*, 2003.

[24] M. Ghallab, "Task execution monitoring by compiles production rules in an advanced multi-sensor robot," in *Proc. Second Int. Symp. Robotics Research*, 1985.

[25] B.J. Glass, W.K. Erickson, and K.J. Swanson, "TEXSYS: A large-scale demonstration of model-based real-time control of space station subsystem," in *Proc. Seventh Int. Conf. on AI Applications*, 1991.

[26] P. Goel, G. Dedeoglu, S.I. Roumeliotis, and G.S. Sukhatme, "Fault detection and identification in a mobile robot using multiple model estimation and neural network," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 2000.

[27] N.J. Gordon, D.J Salmond, and A.F.M Smith, "Novel approach to non-liner/non-Gaussian Bayesian state estimation," *Proc. Inst. Electr. Eng.-F*, vol. 140, no. 2, pp. 107–113, 1993.

[28] J. Gordon and E.H. Shortliffe, "A method for managing evidential reasoning in a hierarchical hypothesis space," *Artif. Intell.*, 1985.

[29] T. Guckenbiehl and G. Schafer-Richter, "SIDIA: Extending prediction-based diagnosis of dynamic models," Price Waterhouse Technology Center, 1990.

[30] W. Hamscher, Modeling Digital Circuits for Troubleshooting, AI, 1991.

[31] P. Harmon, R. Maus, and W. Morrissey, *Expert Systems: Tools and Applications*, New York: Wiley, 1988.

[32] M. Hofbaur and B. Williams, *Mode Estimation of Probabilistic Hybrid Systems*, Hybrid Systems: Computation and Control, 2002.

[33] L.J. Holtzblatt, M.J. Neiberg, and R.L. Piazza, "Temporal reasoning in an ATMS," MITRE Corporation, M91-22, 1991.

[34] F. Hutter and R. Dearden, "The Gaussian particle filter for diagnosis of non-linear systems," in *Proc. Fifth IFAC Symp. on Fault Detection, Supervision, and Safety of Technical Processes*, 2003.

[35] M. Isard and A. Blake, "CONDENSATION: Conditional density propagation for visual tracking," *Int. J. Comput. Vis.*, vol. 29, no. 1, pp. 5–28, 1998.

[36] R. Isermann and P. Balle, "Trends in the application of model-based fault detection and diagnosis of technical processes," *Control Eng. Practice*, 1997.

[37] S.J. Julier and J.K. Uhlmann, "A new extension of the Kalman filter to nonlinear systems," in *Proc. AeroSense: The 11th Int. Symp. on Aerospace/Defence Sensing, Simulation, and Controls*, 1997.

[38] R. Kalman, "A new approach to linear filtering and prediction problems," *J. Basic Eng.*, no. 82, pp. 34–45, 1960.

[39] K. Kanazawa, D. Koller, and S.J. Russel, "Stochastic simulation algorithms for dynamic probabilistic networks," in *Proc. of UAI*, 1995.

[40] G. Kitagawa, "A Monte Carlo filtering and smoothing method for non-Gaussian nonlinear state space models," in *Proc. Second U.S.-Japan Joint Seminar on Statistical Time Series Analysis*, 1993.

[41] X. Koutsoukos, J. Kurien, and F. Zhao, "Monitoring and diagnosis of hybrid systems using particle filtering methods," in *Proc. Int. Symp. on Mathematical Theory of Networks and Systems*, 2002.

[42] J. Kurien and P. Nayak, Back to the Future with Consistency-based Trajectory Tracking, AAAI 2000.

[43] C. Leger, *Darwin2K: An Evolutionary Approach to Automated Design for Robotics*, Norwell: MA: Kluwer, 2000.

[44] U. Lerner, R. Parr, D. Koller, and G. Biswas, "Bayesian fault detection and diagnosis in dynamic systems," in *Proc. 17th Nat. Conf. on Artificial Intelligence*, 2000.

[45] M.L. Littman, "The witness algorithm for solving partially observable Markov decision processes," Brown University, 1994, CS-94-40, Providence, Rhode Island.

[46] J. Liu and R. Chen, "Sequential Monte Carlo methods for dynamic systems, *J. Amer. Statistical Assoc.*, vol. 93, 1998.

[47] S. McIlraith, "Diagnosing hybrid systems: A Bayesian model selection approach," in *Proc. 11th Int. Workshop on Principles of Diagnosis (DX'00)*, June 2000, pp. 140–146.

[48] S. McIlraith, "Explanatory diagnosis: Conjecturing actions to explain observations," in *Proc. Sixth Int. Conf. on Principles of Knowledge Representation and Reasoning*, 1998.

[49] R. Morales-Mendendez, N. de Freitas, and D. Poole, Real-Time Monitoring of Complex Industrial Processes with Particle Filters, NIPS, 2002.

[50] K.J. Aström, "Optimal control of Markov decision processes with incomplete state estimation," *J. Math. Anal. Appl.*, 1965.

[51] L.P. Kaelbling, M.L. Littman, and A.R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artif. Intell.*, vol. 101, 1998.

[52] M. Pitt and N. Shephard, "Filtering via simulation: Auxiliary particle filter," *J. Amer. Statistical Assoc.*, vol. 94, no. 446, 1999.

[53] M. Pitt and N. Shephard, "Filtering via simulation: Auxiliary particle filter," *J. Amer. Statistical Assoc.*, 1999.
[54] A.R. Cassandra, L.P. Kaelbling, and M.L. Littman, "Acting optimally in partially observable stochastic domains," in *Proc. 12fth Nat. Conf. on Artificial Intelligence*, (AAAI), 1994.
[55] A.R. Cassandra, L.P. Kaelbling, and J.A. Kurien, "Discrete Bayesian uncertainty models for mobile-robot navigation," Department of Computer Science Brown University.
[56] R.R. Murphy and D. Hershberger, "Classifying and recovering from sensing failures in autonomous mobile robots," in *Proc. AAAI/IAAI*, 1996.
[57] R.R. Murphy and R. Arkin, "An architecture for action-oriented sensor fusion," in *Proc. IROS*, 1992.
[58] F.R. Noreils, "Integrating error recovery in a mobile robot control system," in *Proc. IEEE Int. Conf. on Robotics and Automation*, May 1990.
[59] F.R. Noreils and R. Chatila, "Plan execution monitoring and control architecture for mobile robots," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 1995.
[60] S.I. Roumeliotis, G.S. Sukhatme, and G.A. Bekey, "Sensor fault detection and identification in a mobile robot," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 1998.
[61] S.I. Roumeliotis, G.S. Sukhatme, and G.A. Bekey, "Fault detection and identification in a mobile robot using multiple-model estimation," in *Proc. 1998 IEEE Int. Conf. on Robotics and Automation*, May 1998, pp. 2223–2228.
[62] D.B. Rubin, *Bayesian Statistics 3, Using the SIR Algorithm to Simulate Posterior Distributions*, London, U.K.: Oxford Univ. Press, 1988.
[63] D.S. Wettergreen, M.B. Dias, B. Shamah, J. Teza, P. Tompkins, C. Urmson, M.D. Wagner, and W.L. Whittaker, "First experiment in sun-synchronous exploration,"in *Proc. Int. Conf. on Robotics and Automation*, 2002.
[64] J.S. Brown, R.R. Burton, and J. de Kleer, "Pedagogical, natural language and knowledge engineering techniques in SOPHIE I, II, and III, *Intelligent Tutoring Syst.*, 1982.
[65] E. Shortliffe, *MYCIN: Computer-Based Medical Consultations*, New York: Elsevier, 1976.
[66] R. Simmons, "Becoming increasingly reliable," in *Proc. Second Int. Conf. on Artificial Intelligence Planning Systems*, 1994.
[67] E.J. Sondik, The optimal control of partially observable Markov Decision Processes, Stanford, 1971.
[68] P. Szolovits and S.G. Pauker, "Categorical and probabilistic reasoning in medical diagnosis," *Artif. Intell.*, 1978.
[69] P. Szolovits and S.G. Pauker, "Categorical and probabilistic reasoning in medical diagnosis revisited," *Artif. Intell.*, 1993.
[70] M.A. Tanner, *Tools for Statistical Inference*, Berlin, Germany: Springer Velag, 1996.
[71] S. Thrun, J. Langford, and V. Verma, Risk Sensitive Particle Filters, Neural Information Processing Systems (NIPS), Dec. 2001.
[72] S. Thrun, D. Fox, and W. Burgard, "Monte Carlo localization with mixture proposal distribution," in *Proc. AAAI Nat. Conf. on Artificial Intelligence*, AAAI 2000.
[73] U. Lerner, R. Parr, D. Koller, and G. Biswas, "Bayesian fault detection and diagnosis in dynamic systems," in *Proc. 17th Nat. Conf. on Artificial Intelligence*, AAAI, 2000.
[74] R. van der Merwe, N. de Freitas, A. Doucet, and E. Wan, The Unscented Particle Filter, Cambridge University Engineering Department, 2000, CUED/F-INFENG/TR 380, Cambridge, England.
[75] V. Verma, J. Langford, and R. Simmons, "Non-parametric fault identification for space rovers," in *Proc. Int. Symp. on Artificial Intelligence and Robotics in Space (iSAIRAS)*, June 2001.
[76] V. Verma, S. Thrun, and R. Simmons, "Variable resolution particle filter," in *Int. Joint Conf. of Artificial Intelligence*, 2003.
[77] V. Verma, G. Gordon, and R. Simmons, "Efficient monitoring for planetary rovers," *Int. Symp. on Artificial Intelligence and Robotics in Space*, 2003.
[78] V. Verma, Anecdotes from rover field operations, RIACS Summer student research program report-NASA Ames research center, 2001.
[79] R. Washington, "On-board real-time state and fault identification for rovers," in *Proc. 2000 IEEE Int. Conf. on Robotics and Automation,* 2000.
[80] B.C. Williams and P. Nayak, "A model-based approach to reactive self-configuring systems," in *Proc. AAAI*, 1996.

Vandi Verma is a Ph.D. student in Robotics at Carnegie Mellon University. Her primary research interests are AI, robotics, and machine learning. Her current research focus is probabilistic reasoning for nonparametric state estimation and control of systems with rare events. Her dissertation work, "Hierarchical Fault Detection and Identification using Particle Filters," focuses on applying these techniques to rover fault detection and classification. During her graduate work she has developed algorithms for navigation, control, and state estimation of a number of robots including Nomad (CMU, Antarctic meteorite search), Lama (LAAS, France), Bullwinkle (CMU, long-range Mars navigation), Hyperion (CMU, Sun-synchronous navigation), K9 (NASA Ames), and the robotic astrobiologist (CMU, life in the Atacama project).

Geoff Gordon is a faculty member in the Center for Automated Learning and Discovery (CALD) at Carnegie Mellon. He is interested in reinforcement learning/planning, statistical models of difficult data (examples include natural-language text and maps of a robot's surroundings), and computational learning theory. He is currently a visiting professor at the Stanford Robotics Lab. Earlier he worked for Burning Glass Technologies, a company that provides intelligent searching and matching software for resumes and job postings, at the Pittsburgh office. Before that, he was a postdoctoral researcher at the AUTON lab in the Robotics Institute and a computer science Ph.D. student at Carnegie Mellon, with advisor Tom Mitchell.

Reid Simmons is a research professor in the School of Computer Science at Carnegie Mellon University. He earned his B.A. degree in 1979 in computer science from SUNY at Buffalo and his M.S. and Ph.D. degrees from MIT in 1983 and 1988, respectively, in the field of artificial intelligence. Since coming to Carnegie Mellon in 1988, his research has focused on developing self-reliant robots that can autonomously operate over extended periods of time in unknown, unstructured environments. This work involves issues of robot control architectures that combine deliberative and reactive control, probabilistic planning and reasoning, monitoring and fault detection, and robust indoor and outdoor navigation. More recently, Dr. Simmons has focused on the areas of coordination of multiple heterogeneous robots, human-robot social interaction, and formal verification of autonomous systems. Over the years, he has been involved in the development of over a dozen autonomous robots.

Sebastian Thrun is the director of the AI Lab at Stanford University and an associate professor of computer science. He pursues research in probabilistic reasoning, robotics, and machine learning with enthusiasm.

Address for Correspondence: Vandi Verma, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213 USA. E-mail: vandi@cs.cmu.edu.