Efficient On-line Fault Diagnosis for Non-Linear Systems

Frank Hutter[*]
Fachbereich Informatik
Technische Universität Darmstadt, Germany
mail@fhutter.de

Richard Dearden
RIACS / NASA Ames Research Center
M.S. 269-3 Moffett Field, CA 94035 USA
dearden@email.arc.nasa.gov

Abstract

Fault diagnosis is a critical task for autonomous operation of systems such as spacecraft and planetary rovers, and must often be performed on-board. Unfortunately, these systems frequently also have relatively little computational power to devote to diagnosis. For this reason, algorithms for these applications must be extremely efficient, and preferably anytime.

In this paper we introduce the *Gaussian particle filter* (GPF), an efficient variant on the particle filtering algorithm for non-linear hybrid systems. Each particle samples a discrete mode and approximates the continuous variables by a multivariate Gaussian that is updated at each time-step using an unscented Kalman filter.

The algorithm is closely related to Rao-Blackwellized Particle Filtering and equally efficient, but is more broadly applicable. We show that given the same computation time GPF performs diagnosis with a significantly lower rate of incorrect diagnoses and with a much lower error on the continuous parameters. We also use the GPF to diagnose data from the K-9 rover at NASA Ames Research Center.

1 Introduction

Fault diagnosis is a critical task for autonomous operation of systems such as spacecraft and planetary rovers. The diagnosis problem is to determine the state of a system over time given a stream of observations of that system. A common approach to this problem is *model-based diagnosis* [2, 3], in which the overall system state is represented as an assignment of a *mode* (a discrete state) to each component of the system. Such an assignment is a possible description of the current state of the system if the set of models associated with the modes is consistent with the observed sensor values. One example of such a system is Livingstone [19], which flew on the Deep Space One spacecraft as part of the Remote Agent Experiment [12] in May 1999. In Livingstone, diagnosis is performed by maintaining a candidate hypotheses (in other systems more than one hypothesis is kept) about the current state of each system component, and comparing the candidate's predicted behaviour with the system sensors. Traditional model-based diagnosis operates on discrete models only, and uses *monitors* to translate continuous sensor readings into discrete values. The monitors are typically only used once the sensor readings have settled on a consistent value, and hence these systems cannot generally diagnose transient events.

For many applications, e.g. planetary rovers, the complex dynamics of the system make reasoning with a discrete model inadequate. This is because too fine a discretization is required to accurately model the system; because the monitors would need global sensor information to discretize a single sensor correctly; and because transient events must be diagnosed. To overcome this we need to reason directly with the continuous values we receive from sensors: Our model needs to be a hybrid system.

A hybrid system consists of a set of discrete *modes*, which represent fault states or operational modes of the system, and a set of continuous variables which model the continuous quantities that affect system behaviour. We will use the term *state* to refer to the combination of these, that is, a state is a mode plus a value for each continuous variable, while the *mode* of a system refers only to the discrete part of the state. In many cases, not all of the hybrid system will be observable. Therefore, we also have an observation function that defines the likelihood of an observation given the mode and the values of the continuous variables. All these processes are inherently noisy, and the representation reflects this by explicitly including noise in the continuous values, and stochastic transitions between system modes. We describe our hybrid model in more detail in Section 2.

There are several challenges to overcome to produce an effective diagnosis algorithm for these types of hybrid models. The algorithm we present here attempts to make progress on all these problems, although primarily on the first four:

Very low prior fault probabilities: Diagnosis problems are particularly difficult for approximation algorithms based on sampling because of the low probabilities of transitions to fault states. Because of those low priors only a very small fraction of the samples moves to a fault state even if this state per-

[*]Frank Hutter was partially supported by a RIACS SSRP student fellowship at NASA Ames Research Center.

fectly explains the system behaviour. This can lead to incorrect diagnoses when no samples move to a fault state at all, even though it has a high posterior probability.

Restricted computational resources: For space applications, computation time is frequently at a premium, and on-board real-time diagnosis is often necessary. For this reason, diagnosis must be as efficient as possible.

Non-linear stochastic transitions and observations: Many algorithms are restricted to linear models with Gaussian noise. Our domains frequently behave non-linearly, so we would prefer an algorithm without this restriction.

Multimodal system behaviour: Even in a single discrete mode, the observations are often consistent with several values for the continuous variables, and so multi-modal distributions appear. For example, when a rover is commanded to accelerate, we are often uncertain about exactly when the command is executed. Different start times lead to different estimates of current speed, and hence a multi-modal distribution. Again, this is a problem for a number of algorithms, particularly for Kalman filters.

High dimensional state spaces: As the number of possible faults grows, the number of discrete modes in the system grows exponentially. The number of samples required to accurately approximate the posterior distribution also grows exponentially with the dimensionality of the continuous state.

Computing exact diagnoses for a probabilistic dynamic model such as the one we describe above is computationally intractable in the general case. There are exceptions the most important of which is solved optimally and efficiently by the Kalman filter [7]. We introduce Kalman filters in Section 2.1. For more general cases a number of authors have proposed approximate inference algorithms [16, 18]. The most general approach proposed is the particle filter [9, 5] which sequentially computes an approximation to the posterior probability distribution of the states of the system given the observations. The posterior distribution is approximated by a set of point samples or *particles*. We discuss particle filters in Section 2.2, for a much more extensive review see [6].

An increasingly commonly used variant on particle filters is the Rao-Blackwellized particle filter (RBPF) [1], which we describe in Section 2.3. Essentially, RBPF combines a particle filter for the discrete system modes with Kalman filters [7] to compute the distribution of the continuous state. This greatly reduces the computational requirements of the algorithm by representing the continuous state more efficiently, but it is restricted in its application to problems in which the differential equations controlling the continuous state variables are linear equations, and all noise is Gaussian. To extend these computationally efficient approaches to non-linear systems we introduce the *Gaussian particle filter*, which keeps samples over the discrete modes and a distribution over the continuous state just as RBPF does, but uses the unscented transformation to keep the distribution updated in non-linear models. We describe the Gaussian particle filter in Section 3, and demonstrate its effectiveness in Section 4. On artificial data, we show that the Gaussian particle filter (GPF) and GPF2, an efficient variant of it significantly outperform basic particle filters and the unscented particle filter. We also show that these filters enable us to do online diagnosis on real data.

2 Hybrid State Estimation

Following [8] and [10], we model the system to be diagnosed as a discrete-time probabilistic hybrid automaton (PHA). A PHA is a tuple $\langle Z, X, Y, F, G, T, P \rangle$, where:

- $Z = z_1, \ldots, z_n$ is the set of discrete modes the system can be in.
- $X = x_1, \ldots, x_m$ is the set of continuous state variables which capture the dynamic evolution of the automaton.
- $Y = Y_c \cup Y_d$ is the set of observable variables. Y_c is the set of continuous observable variables, while Y_d is the set of discrete observable variables, typically commands sent to the system.
- $F = F_1, \ldots, F_n$ is, for each mode z_i the set of discrete time difference equations F_i that describe the evolution of the continuous variables X in that mode. We write $P(X_t|z_{t-1}, x_{t-1})$ for the distribution over X at time t given that the system is in state (z, x) at $t-1$.
- $G = G_1, \ldots, G_n$ is, for each mode, the set of equations governing the relationship between the observational variables Y and the state variables X. We write $P(Y_t|z_t, x_t)$ for the distribution of observations in state (z_t, x_t).
- T is a probabilistic transition function over the discrete modes that specifies $P(Z_t|z_{t-1}, x_{t-1})$, the conditional probability distribution over modes at time t given that the system is in state (z, x) at $t-1$. In some systems, this is independent of the continuous variables: $P(Z_t|z_{t-1}, x_{t-1}) = P(Z_t|z_{t-1})$.
- P is the prior distribution $P(Z_0, X_0)$ over states of the system.

We denote a hybrid state of the system by $s = (z, x)$, which consists of a discrete mode z, and an assignment to the state variables x.

Diagnosis of a hybrid system of this kind is determining, at each time-step, the *belief state* $P(S_t|y_{1:t})$, a distribution that, for each state s, gives the probability that s is the true state of the system, given the observations so far. In principle, belief state tracking is an easy task, which can be performed using the *forward pass* equation:

$$\begin{aligned} P(s_t|y_{1:t}) &= \alpha P(y_t|s_t)\int P(s_t|s_{t-1})P(s_{t-1}|y_{1:t-1})ds_{t-1} \\ &= \alpha P(y_t|z_t,x_t) \\ &\int P(x_t|z_t,x_{t-1})P(z_t|z_{t-1},x_{t-1})P(s_{t-1}|y_{1:t-1})ds_{t-1} \end{aligned}$$

where α is a normalizing constant. Unfortunately, computing the integral exactly is intractable in all but the smallest of problems, or in certain special cases. The most important special case is a unimodal linear model with Gaussian noise. This is solved optimally and efficiently by the Kalman filter (KF). We describe the KF below; then, we weaken the model restrictions and describe algorithms for more general models, such as Particle Filters and Rao-Blackwellized Particle Filters. We end with the most general problem for which we propose the Gaussian Particle Filter.

2.1 Kalman Filters

When the system we want to diagnose has only one discrete mode, linear transition and observation functions for the continuous parameters and Gaussian noise there exists a closed form solution to the tracking problem. In this case, the belief state is a multivariate Gaussian and can be computed incrementally using a *Kalman filter* (KF). At each time-step t the Kalman filtering algorithm updates sufficient statistics $(\mu_{t-1}, \Sigma_{t-1})$, prior mean and covariance of the continuous distribution, with the new observation y_t. We omit details and the Kalman equations here, and refer interested readers to [7].

The Kalman filter is an extremely efficient algorithm. However, in the case of non-linear transformations it does not apply; good approximations are achieved by the *extended Kalman filter* (EKF) and the *unscented Kalman filter* (UKF) with the UKF generally dominating the EKF [17]. Rather than using the standard Kalman filter update to compute the posterior distribution, the UKF performs the following: Given an m-dimensional continuous space, $2m+1$ *sigma points* are chosen based on the a-priori covariance (see [17] for details). The non-linear system equation is then applied to each of the sigma points, and the a-posteriori distribution is approximated by a Gaussian whose mean and covariance are computed from the sigma points. The mean is set to the weighted mean of the transitioned sigma points and the covariance is taken to be the sum of the weighted squared derivations of the transitioned sigma points from the mean. This unscented Kalman filter update yields an approximation of the posterior whose error depends on how different the true posterior is from a Gaussian. For linear and quadratic transformations, the error is zero.

```
1. For $N$ particles $p^{(i)}$, $i = 1, \ldots, N$, sample discrete modes $z_0^{(i)}$, from the prior $P(Z_0)$.
2. For each particle $p^{(i)}$, sample $x_0^{(i)}$ from the prior $P(X_0|z_0^{(i)})$.
3. for each time-step $t$ do
   (a) For each particle $p^{(i)} = (z_{t-1}^{(i)}, x(i)_{t-1})$ do
       i. Sample a new mode:
          $\hat{z}_t^{(i)} \sim P(Z_t|z_{t-1}^{(i)})$.
       ii. Sample new continuous parameters:
          $\hat{x}_t^{(i)} \sim P(X_t|\hat{z}_t^{(i)}, x_{t-1}^{(i)})$.
       iii. Compute the weight of particle $\hat{p}^{(i)}$:
          $w_t^{(i)} \leftarrow P(y_t|\hat{z}_t^{(i)}, \hat{x}_t^{(i)})$.
   (b) Resample $N$ new samples $p^{(i)}$ where: $P(p^{(i)} = \hat{p}^{(k)}) \propto w_t^{(k)}$
```

Figure 1: The particle filtering algorithm.

2.2 Particle Filters

While the success of the above approaches depend on how strongly the belief state resembles a multivariate Gaussian, the *particle filter* (PF) [9] is applicable regardless of the underlying model. A particle filter is a Markov chain Monte Carlo algorithm that approximates the belief state using a set of samples (particles), and keeps the distribution updated as new observations are made over time. The basic PF algorithm is shown in Figure 1. To update the belief distribution given a new observation, the algorithm operates in three steps as follows:

The Monte Carlo step: This step considers the evolution of the system over time. It uses the stochastic model of the system to generate a possible future state for each sample. In our hybrid model (and Figure 1), this is performed by sampling a discrete mode, and then the continuous state given the new mode.

The reweighting step: This corresponds to conditioning on the observations. Each sample is weighted by the likelihood of seeing the observations in the (updated) state represented by the sample. This step leads samples that predict the observations well to have high weight, and samples that are unlikely to generate the observations to have low weight.

The resampling step: To produce a uniformly weighted posterior, we then resample a set of uniformly weighted samples from the distribution represented by the weighted samples. In this resampling the probability that a new sample is a copy of a particular sample s is proportional to the weight of s, so high-weight samples may be replaced by several samples, and low-weight samples may disappear.

At any time t, the PF algorithm approximates the true posterior belief state given observations $y_{1:t}$ by a set of samples (or particles):

$$\begin{aligned} P(Z_t, X_t|y_{1:t}) &\approx \hat{P}(Z_t, X_t|y_{1:t}) \\ &= \frac{1}{N}\sum_{i=1}^{N} w_t^{(i)} \delta_{(Z_t,X_t)}((z_t^{(i)}, x_t^{(i)})) \end{aligned}$$

where $w_t^{(i)}$, $z_t^{(i)}$ and $x_t^{(i)}$ are weight, discrete mode and continuous parameters of particle $p^{(i)}$ at time t, N is the number of samples, and $\delta_x(y)$ denotes the Dirac delta function.

Particle filters have a number of properties that make them a desirable approximation algorithm for diagnosis. As we said above, unlike the Kalman filter, they can be applied to non-linear models with arbitrary prior belief distributions. They are also *contract anytime* algorithms, meaning that if you specify in advance how much computation time is available, a PF algorithm can estimate a belief distribution in the available time—by changing the number of samples, you trade off computation time for the quality of the approximation. In fact, the computational requirements of a particle filter depend *only* on the number of samples, not on the complexity of the model.

Unfortunately, as we said in the introduction, diagnosis problems have some characteristics that make standard particle filtering approaches less than ideal. In particular, on-board diagnosis for applications such as spacecraft and planetary rovers must be performed using very limited computational resources, and transitions to fault modes typically have very low probability of occurring. This second problem leads to a form of *sample impoverishment*, in which modes with a non-zero probability of being the actual state of the system contain no samples, and are therefore treated by the particle filter as having zero probability. This is particularly a problem for diagnosis, because these are exactly the states for which we are most interested in estimating the likelihood. There have been a few approaches to tackling this issue, most notably [4] and [14].

Another traditional problem of particle filters is that the number of samples needed to cope with high dimensional continuous state spaces is enormous. Especially in the case of high noise levels and widespread distributions, approximations via sampling do not yield good results. If it is possible to represent the continuous variables in a compact way, e.g. in the form of sufficient statistics, this generally helps by greatly reducing the number of particles needed. In the next section, we introduce one instance of this, the highly efficient Rao-Blackwellized Particle Filter which only samples the discrete modes and propagates sufficient statistics for the continuous variables.

2.3 Rao-Blackwellized Particle Filters

Recent work on *Rao-Blackwellized Particle Filtering* (RBPF) [1, 11] has focused on combining PFs and KFs for tracking linear multimodal systems with Gaussian noise. In this kind of model, the belief state is a mixture of Gaussians. Rather than sampling a complete system state, in RBPF for hybrid systems, one combines a Particle Filter that samples the discrete modes z_t, and a Kalman Filter for each discrete mode $z_t \in Z$ that propagates sufficient statistics $(\mu_t^{(i)}, \Sigma_t^{(i)})$ for the continuous parameters x_t. The algorithm is shown in Figure 2. At each time-step t, first, the discrete mode is sampled according to the transition prior. Then, for each particle $p^{(i)}$ a Kalman filter is called to compute the prior mean $\hat{y}_{t|t-1}^{(i)}$ and covariance $\hat{S}_t^{(i)}$ of the observation and update the mean $\mu_t^{(i)}$ and covariance $\Sigma_t^{(i)}$ for the continuous parameters. The variable $\theta(z_t^{(i)})$ denotes the parameters of the Kalman Filter belonging to mode $z_t^{(i)}$. Finally, the particle weight is computed as the observation probability $P(y_t|\hat{y}_{t|t-1}^{(i)}, \hat{S}_t^{(i)})$ of y_t given the prior observation mean and covariance. As in regular Particle Filtering, a resampling step is necessary to prevent particle impoverishment.

```
1. For $N$ particles $p^{(i)}, i = 1, \dots, N$, sample discrete
   modes $z_0^{(i)}$, from the prior $P(Z_0)$.
2. For each particle $p^{(i)}$, set $\mu_0^{(i)}$ and $\Sigma_0^{(i)}$ to the prior
   mean and covariance in state $z_0^{(i)}$.
3. For each time-step $t$ do
   (a) For each $p^{(i)} = (z_{t-1}^{(i)}, \mu_{t-1}^{(i)}, \Sigma_{t-1}^{(i)})$ do
       i. Sample a new mode:
          $\hat{z}_t^{(i)} \sim P(Z_t|z_{t-1}^{(i)})$.
       ii. Perform Kalman update using parameters from mode $\hat{z}_t^{(i)}$:
          $(\hat{y}_{t|t-1}^{(i)}, \hat{S}_t^{(i)}, \hat{\mu}_t^{(i)}, \hat{\Sigma}_t^{(i)})$
              $\leftarrow KF(\mu_{t-1}^{(i)}, \Sigma_{t-1}^{(i)}, y_t, \theta(z_t^{(i)}))$.
       iii. Compute the weight of particle $\hat{p}^{(i)}$:
          $w_t^{(i)} \leftarrow P(y_t|\hat{y}_{t|t-1}^{(i)}, \hat{S}^{(i)}) = N(y_t; \hat{y}_{t|t-1}^{(i)}, \hat{S}^{(i)})$.
   (b) Resample as in step 3.(b) of the PF algorithm
       (see Figure 1).
```

Figure 2: The RBPF algorithm.

As shown in [11], it is possible in Rao-Blackwellized Particle Filtering to sample the discrete modes directly from the posterior. It is also possible to resample *before* the transition according to the expected posterior weight distribution such that those particles get multiplied which are likely to transition to states of high confidence. These improvements result in an even more efficient algorithm called RBPF2 [11].

3 Non-Linear Estimation

Since RBPF uses a KF for its continuous state estimation, it is restricted to linear problems with Gaussian noise. Many of the problems we are interested in do not have these properties. To overcome this, we propose the *Gaussian particle filter* (GPF), an efficient variant of particle filtering for non-linear hybrid models that is conceptually closely related to RBPF. In general hybrid systems, there is no tractable closed-form solution for the continuous variables, so we cannot maintain sufficient statistics with every sample. It is however possible to propagate an approximation of the continuous variables. We sample the mode as usual and for every particle update a Gaussian approximation of the continuous parameters using an unscented Kalman filter. Since the unscented Kalman filter only approximates the true posterior distribution, the GPF is a biased estimator in non-linear models; omitting the sampling of continuous variables however greatly reduces the estimator's variance. However, PFs with a finite number of samples also yield a biased estimator and the variance of the estimate is greatly reduced in the GPF.

The GPF algorithm is very similar to the RBPF algorithm presented in Figure 2. In both of these algorithms particle $p^{(i)}$ represents the continuous variables with a multivariate Gaussian $N(\mu_t^{(i)}, \Sigma_t^{(i)})$. In the case of linear models and RBPF, this Gaussian is a sufficient statistic, in the case of non-linear models and GPF, it is an approximation. In the algorithm, the only change is in line 3.(a)ii of Figure 2, which is replaced by:

```
3.a(ii) Perform an unscented Kalman update using parameters from mode $\hat{z}_t^{(i)}$:

    $(\hat{y}_{t|t-1}^{(i)}, \hat{S}_t^{(i)}, \hat{\mu}_t^{(i)}, \hat{\Sigma}_t^{(i)})$
        $\leftarrow UKF(\mu_t^{(i)}, \Sigma_t^{(i)}, y_t, \theta(z_t^{(i)}))$
```

This change is due to the non-linearity of transition and/or observation function. A Kalman update is simply not possible, but a good approximation is achieved with an unscented Kalman filter. The approximation of continuous variables in the GPF is a mixture of Gaussians rather than the set of samples as in a PF. Since the expressive power of every particle is higher, fewer particles are needed to achieve the same approximation accuracy. This more than offsets the small additional computational cost per sample. Furthermore, this compact approximation is likely to scale smoothly with an increase in dimensionality.

Like RBPF, the GPF can be improved by sampling directly from the posterior distribution and resampling *before* the transition. We call the resulting algorithm GPF2 and detail it in Figure 3. For each particle, before actually sampling a discrete mode, we look at each possible mode m, update our approximations of the continuous parameters assuming we had sampled m, and compute the observation likelihood for those approximations. This and the transition prior give the posterior probability of transitioning to m. Then for each particle we sample a new discrete mode from the posterior we computed for it.

At each time-step t, for every particle $p^{(i)}$, first we enumerate each possible successor mode m, i.e. each mode $m \in Z$ such that $P(m|z_{t-1}^{(i)}) > 0$. For each m, we perform an unscented Kalman update, and compute analytically the observation likelihood $P(y_t|m, \mu_{t-1}^{(i)}, \Sigma_{t-1}^{(i)}) = P(y_t|y_{t|t-1}^{(i,m)}, S_t^{(i,m)})$. Then, we compute the unnormalized posterior probability $Post(i,m)$ of transitioning to mode m with particle $p^{(i)}$; this is given simply by the product of the transition prior to m and the observation likelihood in m. Next we compute the weight of each particle $\hat{p}^{(i)}$ as the sum of the posterior probabilities of it's successor modes and resample N particles according to this weight distribution. Note, that $Post(i,m)$, $\mu_t^{(i,m)}$ and $\Sigma_t^{(i,m)}$ also need to be resampled, i.e. when particle $p^{(i)}$ is sampled to be particle $\hat{p}^{(k)}$, then $Post(i,m) \leftarrow \widehat{Post}(k,m)$, $\mu_t^{(i,m)} \leftarrow \hat{\mu}_t^{(k,m)}$ and $\Sigma_t^{(i,m)} \leftarrow \hat{\Sigma}_t^{(k,m)}$ for all m.

Finally, for every particle $p^{(i)}$, a successor mode m is sampled according to the posterior probability; this mode is used as $z_t^{(i)}$; $\mu_t^{(i)}$ and $\Sigma_t^{(i)}$ are set to the already computed value $\mu_t^{(i,m)}$ and $\Sigma_t^{(i,m)}$.

GPF2 only differs from the RBPF2 algorithm in that it is calling an unscented Kalman filter update instead of a Kalman update due to the non-linear character of the transformations. It is a very efficient algorithm for state estimation on non-linear models with transition and observation functions that transform a Gaussian distribution to a distribution that's close to a Gaussian. Very low fault priors are handled especially gracefully by GPF2 since it samples the discrete modes from their true posterior distribution. When there is strong enough evidence the fault will be detected regardless of how low the prior is.

4 Experiments

We performed experiments on a simple model of the suspension system of the K-9 rover at NASA Ames Research Center. K-9 is a six wheeled rover with a rocker-bogey suspension, and we model the suspension's response to driving over rocks and other obstacles to anticipate situations where the rover's scientific instruments could collide with an obstacle, or where the rover could become "high-centered" on a rock. The model has six discrete modes and six continuous variables, two of which are observable. The continuous parameters follow non-linear trajectories (trigonometric functions) in three of the modes, and are linear in the others. As well as tracking the discrete mode of the system, we are also interested in estimating the values of one of the unobservable state variables, which corresponds to the height of the obstacle being traversed, and hence to the clearance between the rover and the obstacle.

Figure 4 shows the rate of state estimation errors for

```
1. For $N$ particles $p^{(i)}$, $i = 1, \ldots, N$, sample discrete modes $z_0^{(i)}$, from the prior $P(Z_0)$.
2. For each particle $p^{(i)}$, set $\mu_0^{(i)}$ and $\Sigma_0^{(i)}$ to the prior mean and covariance in state $z_0^{(i)}$.
3. For each time-step $t$ do
   (a) For each $p^{(i)} = (z_{t-1}^{(i)}, \mu_{t-1}^{(i)}, \Sigma_{t-1}^{(i)})$ do
       i. For each possible successor mode $m \in succ(z_{t-1}^{(i)})$ do
          A. Perform unscented Kalman update using parameters from mode $m$:
             $(\hat{y}_{t|t-1}^{(i,m)}, \hat{S}_t^{(i,m)}, \hat{\mu}_t^{(i,m)}, \hat{\Sigma}_t^{(i,m)}) \leftarrow UKF(\mu_{t-1}^{(i)}, \Sigma_{t-1}^{(i)}, y_t, \theta(m))$.
          B. Compute posterior probability of mode $m$ as:
             $\widehat{Post}(i,m) \leftarrow P(m|z_{t-1}^{(i)}, y_t)$
                              $= P(m|z_{t-1}^{(i)}) N(y_t; y_{t|t-1}^{(i,m)}, S_{t|t-1}^{(i,m)})$.
       ii. Compute the weight of particle $\hat{p}^{(i)}$: $w_t^{(i)} \leftarrow \sum_{m \in succ(z_{t-1}^{(i)})} \widehat{Post}(i,m)$
   (b) Resample as in step 2.(b) of the PF algorithm (see Figure 1) (also resample $Post$, $\mu_t$ and $\Sigma_t$).
   (c) For each particle $p^{(i)}$ do
       i. Sample a new mode:
          $m \sim P(Z_t|z_{t-1}^{(i)}, y_t)$.
       ii. Set $z_t^{(i)} \leftarrow m$, $\mu_t^{(i)} \leftarrow \mu_t^{(i,m)}$ and $\Sigma_t^{(i)} \leftarrow \Sigma_t^{(i,m)}$.
```

Figure 3: The GPF2 algorithm.

the GPF, GPF2 and traditional particle filters, as well as the unscented particle filter, discussed below. The model was used to generate data for which ground truth was available as to the true values of the mode and continuous variables, and the algorithms were applied to this artificial data. The diagnosis of every filter is taken to be the maximum a posteriori (MAP) estimate for the discrete modes; we define a discrepancy between this MAP estimate and the real discrete mode as an error. Figure 4 shows the error rates ($\frac{\#diagnosis\ errors}{\#time\ \ steps}$) achieved by the algorithms with different numbers of samples; the x-axis is the CPU time the algorithms needed for the computation. The graph shows that GPF is a better approximation than PF given the same computing resources, particularly as the number of samples increases and the discrete states become adequately populated with samples. GPF2 is considerably slower per sample but its approximation is superior to PF or GPF.

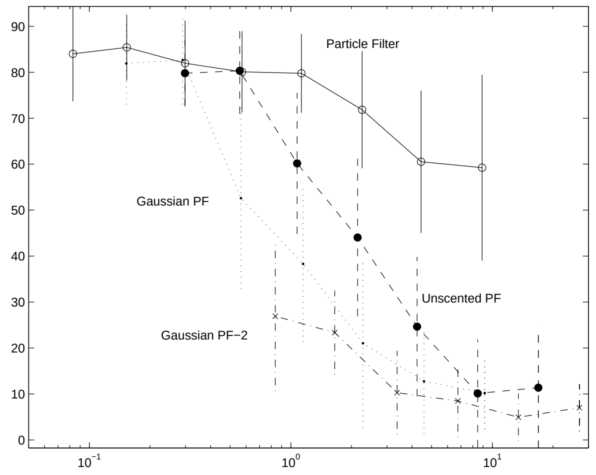

Figure 4: Performance for the GPF, the GPF when sampling from the posterior, the UPF, and traditional particle filters. Estimation based on 50 runs.

We also compare our results with the *unscented particle filter* (UPF) of [15]. The GPF and UPF have a number of similarities. Both use a set of particles each of which performs an unscented Kalman update at every time step. In UPF, the Kalman update approximation $N(mu_t, \Sigma_t)$ of the posterior is used as a proposal for the particle filter, in GPF this approximation is used as the filter result. By sampling from the continuous distribution $N(mu_t, \Sigma_t)$, the unscented particle filter introduces extra variance on its estimation. The GPF is giving up theoretical convergence in order to reduce this variance. In online diagnosis, there are cases where we can only use a few particles for diagnosis. In this case, a low variance is clearly more important than theoretical convergence. Moreover, in practice the UPF is a biased estimator as well since it can only use a finite number of particles.

In some cases we are not only interested in diagnosing the discrete modes, but also the continuous parameters of a hybrid system. As an example we might want to know the steepness of a hill, or the size of a rock we are driving over.

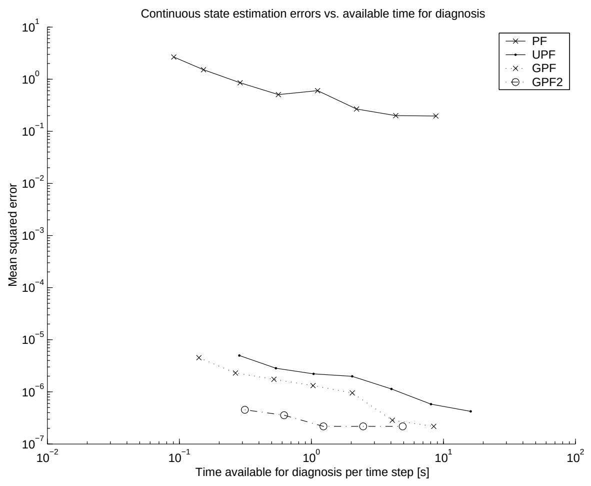

Figure 5: Mean squared errors of the four algorithms, averaged over 50 runs. Note the logarithmic scale for both the X- and Y- axes. At real time (ca. 1/3s per time step), the MSE of GPF2 is about six times lower than of GPF, ten times lower than that for UPF and 10^6 times lower than for PF.

In other applications, these continuous parameters might be crucial, particularly when diagnosing sensor defects and calibration problems. Figure 5 shows the mean squared error (MSE) of the PF, UKF, GPF and GPF2 algorithms on artificial data where ground truth is available. The Y-scale is logarithmic to fit the extreme differences between standard particle filters and the other algorithms. With a mean squares estimation error of the continuous parameters which is about 10^6 times higher than that of GPF2, it is clearly outperformed. The difference between UPF, GPF and GPF2 is less significant, although still a factor of ten between UPF and GPF2.

In our experiments there is little difference between the results of GPF and UPF. GPF is generally faster by a constant factor since it does not need to sample the continuous state, and the weight computation is faster. We would expect the UPF to yield better results when the shape of the posterior distribution is very different from a Gaussian and would expect the GPF to do better when there is a big posterior covariance Σ_t such that the sampling introduces high variance on the estimate. In this case, the UPF will need more particles to yield the same results. Since neither of these conditions applies in our domain, both algorithms show similar performance, with GPF being slightly faster.

We also applied the GPF to real data from the K-9 rover. In Figure 6, we show the two observed variables, differential angle (Y2) and bogey angle (Y1) as well as the discrete mode estimates PF and GPF2 yield. State 1 represents flat driving, state 2 driving over a rock with the front wheel, state 3 with the middle wheel and state 4 with the rear wheel. State 5 represents the rock being between the front and the middle wheel and state 6 between the middle and the rear wheel. There is no ground truth for this data, although it is fairly clear from the data when the rover drives over the three rocks. Both filters successfully detect the rocks, but the GPF2 detects all of the rocks before PF detects them. For the third rock in the data, GPF2 correctly identifies that the back wheel passed over the rock, while the particle filter only tracks the first two wheels. Again, we only show the most probable mode at each time-step in the figure.

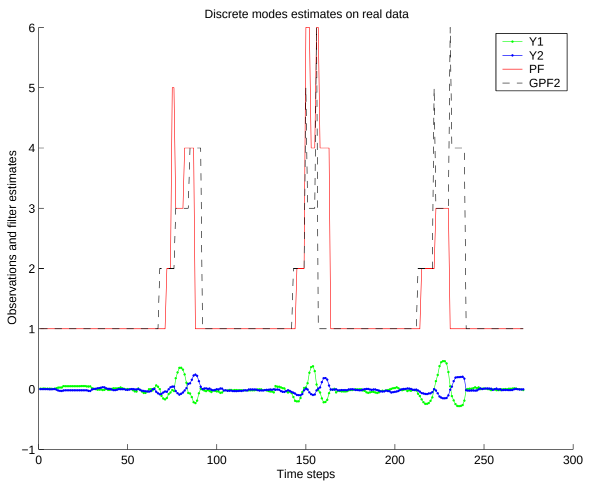

Figure 6: Discrete mode estimates on real data.

5 Conclusions and Future Work

We are currently testing these algorithms on larger models, which should better show the performance gain from GPF over the standard particle filter. We are developing a model of the entire K-9 rover locomotion system, and plan to use these algorithms as part of our diagnosis effort for that system.

The most significant challenge for diagnosis on-board a planetary rover is limited computational resources. Diagnosis is one small part of the overall rover software, and gets a small percentage of the rover's computational resources. Particle filter-based approaches are very attractive because of their anytime properties. The GPF algorithm performs much better than the standard PF with small numbers of samples because it increases the representational power of each sample. We anticipate that GPF will be very useful for on-board diagnosis on the K-9 rover where models can't be easily linearised, and even when the models are linear, as it performs very comparably with RBPF, the best particle filter variant for this case.

Sampling from the true posterior distribution (the GPF2 variant) considerably outperforms GPF in terms of diagnosis errors, even with just a single particle. However, this comes at a large computational cost due to the expense of enumerating the possible modes of the system, and this cost will only worsen as system complexity grows. This may make the algorithm too expensive for on-board diagnosis on the rover, especially as we're generally interested in more than just the most likely diagnosis, and therefore we need a number of samples in the filter. If it is too expensive for general use, it may be useful in situations where an accurate diagnosis is extremely important, and the rover is prepared to spend additional computational resources to ensure that the diagnosis is correct. In this scenario, we might use the GPF algorithm most of the time, but when decisions have to be made that depend crucially on the rover state, GPF2 can be used to refine the current state estimate.

We are also currently investigating efficient variants of GPF2 that do not need to compute the posterior probability for every successor mode. When we consider a system with N faults with very low priors, we conjecture it to be sufficient to consider only a subset of all the 2^N possible successor modes for each particle. One possible approach is to assume that only one or two faults happen at the same discrete time step. This would reduce the number of possible successor modes to N or N^2. The number of faults that are active at any given time would not be bounded, only the number of faults that actually occur at the same time step. Another alternative would be to use a more abstract diagnosis system—possibly even a traditional discrete approach—to reduce the set of possible successor states.

Furthermore, the close relationship between Rao-Blackwellized particle filtering and the Gaussian particle filter opens the possibility for hybrid filters employing the more efficient RBPF algorithm in modes with linear equations and the more general GPF in modes with non-linear equations. The identical representation of the belief state makes it possible to switch algorithms without any computational overhead.

Finally, we are looking at other particle filter-based algorithms such as [13] that divide the system into independent parts, perform diagnoses of each part, and then combine the results into a global diagnosis. Factoring the diagnosis in this way is another important tool for making on-board hybrid diagnosis possible.

References

[1] Nando de Freitas. Rao-Blackwellised particle filtering for fault diagnosis. In *IEEE Aerospace*, 2002.

[2] Johann de Kleer and Brian C. Williams. Diagnosing multiple faults. In Matthew L. Ginsberg, editor, *Readings in Nonmonotonic Reasoning*, pages 372–388. Morgan Kaufmann, Los Altos, California, 1987.

[3] Johann de Kleer and Brian C. Williams. Diagnosis with behavioral modes. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1324–1330, 1989.

[4] Richard Dearden and Dan Clancy. Particle filters for real-time fault detection in planetary rovers. In *Proceedings of the Thirteenth International Workshop on Principles of Diagnosis*, pages 1–6, Semmering, Austria, 2002.

[5] A. Doucet. On sequential simulation-based methods for Bayesian filtering. Technical Report CUED/F-INFENG/TR.310, Department of Engineering, Cambridge University, 1998.

[6] Arnaud Doucet, Nando De Freitas, and Neil Gordon, editors. *Sequential Monte Carlo in Practice*. Springer-Verlag, 2001.

[7] M. S. Grewal and A. P. Andrews. *Kalman Filtering: Theory and Practice*. Prentice Hall, 1993.

[8] Michael W. Hofbaur and Brian C. Williams. Hybrid diagnosis with unknown behaviour modes. In *Proceedings of the Thirteenth International Workshop on Principles of Diagnosis*, pages 97–105, Semmering, Austria, 2002.

[9] M. Isard and A. Blake. CONDENSATION: Conditional density propagation for visual tracking. *International Journal of Computer Vision*, 1998.

[10] Uri Lerner, Ron Parr, Daphne Koller, and Gautam Biswas. Bayesian fault detection and diagnosis in dynamic systems. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence*, 2000.

[11] Ruben Morales-Menendez, Nando de Freitas, and David Poole. Real-time monitoring of complex industrial processes with particle filters. In *Neural Information Processing Systems (NIPS)*, 2002.

[12] Nicola Muscettola, Pandu Nayak, Barney Pell, and Brian Williams. Remote agent: To boldly go where no ai system has gone before. *aij*, 103(1–2), August 1998.

[13] Brenda Ng, Leonid Peshkin, and Avi Pfeffer. Factored particles for scalable monitoring. In *Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence*, Edmonton, 2002.

[14] Sebastian Thrun, John Langford, and Vandi Verma. Risk sensitive particle filters. In *Neural Information Processing Systems (NIPS)*, December 2001.

[15] Rudolph van der Merwe, Nando de Freitas, Arnaud Doucet, and Eric Wan. The unscented particle filter. In *Advances in Neural Information Processing Systems 13*, Nov 2001.

[16] V. Verma, J. Langford, and R. Simmons. Non-parametric fault identification for space rovers. In *International Symposium on Artificial Intelligence and Robotics in Space (iSAIRAS)*, 2001.

[17] E. Wan and R. van der Merwe. The unscented kalman filter for nonlinear estimation. In *Proc. of IEEE Symposium 2000*, Lake Louise, Alberta, Canada, 2000.

[18] Rich Washington. On-board real-time state and fault identification for rovers. In *Proceedings of the IEEE International Conference on Robotics and Automation*, April 2000.

[19] Brian C. Williams and P. Pandurang Nayak. A model-based approach to reactive self-configuring systems. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence and Eighth Innovative Applications of Artificial Intelligence Conference*, pages 971–978, Portland, Oregon, 1996. AAAI Press / The MIT Press.