AIAA Guidance, Navigation, and Control Conference and Exhibit
15 - 18 August 2005, San Francisco, California

AIAA 2005-6484

Check for updates

Particle Filters as Exploration Tools for Autonomous Rovers

Jeffrey M. Sullivan* Mark Campbell† Hod Lipson‡

We present a method for applying particle filters to the problem of search and exploration for autonomous robots. Our approach adaptively controls the resampling stage to prevent premature convergence. Tuning filters is appropriate, and useful, when the subject system is fixed (like a dynamics filter for a particular satellite), but autonomous exploration systems do not fall in to this category. Since they involve traversing and measuring unknown and unpredictable environments, tuning filters for search and exploration is impractical at best. Using a self-tuning filter also adds robustness to faults, which we demonstrate with an example.

We compare our new approach to prior particle filtering algorithms in the context of autonomous rovers using sensors to find and approach objects of interest. To facilitate this comparison, we propose a number of simple control strategies that take advantage of the particle filter structure to guide rovers towards areas of expected high payoff. We also introduce a new metric for filter diversity, termed the condensed number of effective particles, to more accurately reflect the amount of useful information maintained by the different algorithms.

Downloaded by Columbia University on November 3, 2026 | http://arc.aiaa.org | DOI: 10.2514/6.2005-6484

I. Introduction

Past work in the area of search and exploration concentrates on two different goals. The first goal is simultaneous localization and mapping (SLAM) and the second is intelligence, search, and reconnaissance (ISR). SLAM approaches include evidence grids,[1] range scan matching,[2] hierarchical maps,[3] and many other methods. The goal of SLAM is to explore an environment, produce an adequate map of that environment, and maintain an accurate position of the explorer in relation to the whole environment.

ISR approaches focus on gathering information about specific points, objects, or areas and include such techniques as Kalman filtering,[4] information theoretic methods,[5] and a number of stochastic methods.[6,7] These methods concentrate on minimizing the uncertainty of, or maximizing the information in, the state estimates corresponding to objects of interest in an environment. This usually involves a mix of sensor management[8] and path planning[9] to get the most accurate 'picture' of the environment.

Particle filters are a type of stochastic method, used for ISR tasks, and developed to estimate complex probability distributions.[10,11] This filtering technique is chosen when the distribution of states is not Gaussian or when the dynamics are non-linear; it can approximate any distribution (given enough particles) and can deal with "black-box" dynamics. Particle filters consume a constant amount of resources as the explored space expands, deal well with non-linearities, can concurrently maintain multiple hypotheses, and have explicit tradeoffs between computation time and accuracy.

In the following pages, we propose two particle filter improvements to increase robustness. One is a new metric, the condensed number of effective particles, that captures the two problems of degeneracy and sample impoverishment and allows the filtering algorithm to take appropriate corrective measures. The other is an improved roughening[12] step that reduces impoverishment adaptively, applying context-dependent artificial noise to the resampling process, instead of the standard globally determined level of noise.

We also begin a study of how particle filters can be used to make decisions and inform controllers. Our domain of interest requires that we not only gather and maintain accurate information about the environment, but that we also be able to act on that information in pursuit of one or more goals. Since particle filters are used precisely when a standard mean/covariance estimate is inappropriate, new control methods need to be developed that can deal with entire probability distributions as inputs.

*PhD student, Mechanical & Aerospace Engineering, Cornell University, Ithaca NY, graduate student member
†Associate Professor, Mechanical & Aerospace Engineering, Cornell University, Ithaca NY, Senior Member
‡Assistant Professor, Mechanical & Aerospace Engineering, Cornell University, Ithaca NY

Copyright © 2005 by Jeffrey M. Sullivan. Published by the American Institute of Aeronautics and Astronautics, Inc., with permission.

II. A Model of Unmanned Exploration

Our domain of interest is that of unmanned exploration. Our developments have concentrated on the idea of a rover exploring a static environment in search of measurable items of interest. Vehicle control is established through the discrete dynamics model

$$\begin{aligned} \mathbf{x}^1_{k+1} &= f(\mathbf{x}^1_k, \mathbf{c}_k) \\ \begin{bmatrix} x \\ y \\ \theta \\ v \end{bmatrix}_{k+1} &= \begin{bmatrix} x + v\delta_T \cdot \cos(\theta) \\ y + v\delta_T \cdot \sin(\theta) \\ \theta \\ v \end{bmatrix}_k + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & \delta_\theta \\ \delta_v & 0 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}_k \end{aligned} \tag{1}$$

where $\mathbf{c}$ is the control vector and the δ_i are simulation parameters. The state of the vehicle is represented by $\mathbf{x}^1$ whose elements are x-position, y-position, heading, and forward velocity.

While moving through the environment, the vehicle monitors the position of an object using its sensors and a particle filter. We use a bearing sensor that will measure, to within some uncertainty, the relative bearing between the object location and the current vehicle heading. The model for locating one object is

$$\mathbf{z}_k = h(\mathbf{x}_k) \sim \mathcal{N}(\psi - \theta, J) \tag{2}$$

where ψ is the angle of the vector from the vehicle location to the object location. $\mathcal{N}(x, J)$ is a normal distribution with mean x and covariance matrix J. While the distribution of bearing measurements is Gaussian, the underlying distribution of the object location, which is the true state of interest, is not. The state of the object $\mathbf{x}^2$ consists of its x- and y-position. For this work, the object was static and it is assumed that this model is known to the observing vehicle.

This would make the full dynamics for a one vehicle, one object system

$$\begin{bmatrix} \mathbf{x}^1 \\ \mathbf{x}^2 \end{bmatrix}_{k+1} = \begin{bmatrix} f(\mathbf{x}^1_k, \mathbf{c}_k) \\ \mathbf{x}^2_k \end{bmatrix} \tag{3}$$

$$\mathbf{z}_k = h(\mathbf{x}_k) \tag{4}$$

and we assume that the vehicle state is known. The dynamics of the object are linear (the identity matrix), but the measurement is non-linear in the state variables. It is exactly this sort of problem that particle filters were designed for, since no filter producing a single estimate and covariance can capture the information contained in the underlying distribution of the object position (generally wedge-shaped for bearing sensors).

III. Particle Filter Background

Here, we will quickly outline the basic particle filter. Detailed derivations can be found in the references.[13]

A. Motivation for Particle Filtering

The goal of particle filter estimation is to compute an approximation to the probability distribution that represents the outcome of a particular process. This process is typically a dynamic system of the form $\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{v}_k)$ where $\mathbf{v}_k$ is an unknown noise vector and $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ describes how the system $\mathbf{x}$ changes over time. There is usually an associated measurement function $\mathbf{z}_k = h(\mathbf{x}_k, \mathbf{u}_k)$ that maps noisy measurements from the true state $\mathbf{x}_k$.

When the mappings f, h are linear and the noises $\mathbf{v}, \mathbf{u}$ are Gaussian with known parameters, then the optimal solution is a Kalman Filter.[4] When either of the mappings are nonlinear and the noises Gaussian, several solutions exist. The most common is the Extended Kalman Filter for when f can be approximated well by a linearization.[4] More recently, the Sigma Point filter (also known as the Unscented Kalman Filter) was derived to propagate discrete points through the non-linearities of f, without the need for linearization.[14]

Another common choice, one that can accomodate non-Gaussian distributions, is the particle filter. The particle filter represents any arbitrary probability distribution through the use of a finite set of points, called particles. Each particle $\mathbf{p}_i$ has an associated weight w_i that determines its importance to representing the underlying distribution.

Downloaded by Columbia University on November 3, 2026 | http://arc.aiaa.org | DOI: 10.2514/6.2005-6484

As the number of particles grows, so does the accuracy of the approximation. Unlike the Sigma Point filter, the particle filter sample points are not deterministic and are not reduced to a mean and covariance at every step, thus retaining more information about the true distribution.

B. Basic Particle Filter Algorithm and Notation

A particle filter P is a set of N particles $\mathbf{p}_i$ and N weights w_i

$$P = \{(\mathbf{p}_i, w_i) \mid i \in [1, N],\ \mathbf{p}_i \in \mathbb{R}^n,\ w_i \in \mathbb{R}\}. \tag{5}$$

When initialized at time $k = 0$ the particles, $\mathbf{p}_i$, are drawn from the prior distribution (the best available estimate of $\mathbf{x}_0$, defined as $\mathbb{P}_0(\mathbf{x})$) and $w_i = 1/N\ \forall\ i$.

A particle filter is used to approximate the underlying distribution at any point in time using the particles and associated weights. The probability that a given point in the distribution represents the true state is calculated as

$$\mathbb{P}(\mathbf{x}) \approx \sum_{i=1}^{N} w_i \delta(\mathbf{x} - \mathbf{p}_i) \text{ where } \delta(\mathbf{y}) = \begin{cases} 1 & \mathbf{y} = \mathbf{0} \\ 0 & \mathbf{y} \neq \mathbf{0} \end{cases}. \tag{6}$$

Then, given a measurement $\mathbf{z}_k$, the weight of each particle can be updated based on the conditioned sensor measurement distribution as

$$w_{i,k} \propto w_{i,k-1} \frac{\mathbb{P}(\mathbf{z}_k|\mathbf{p}_{i,k})\mathbb{P}(\mathbf{p}_{i,k}|\mathbf{p}_{i,k-1})}{q(\mathbf{p}_{i,k}|\mathbf{p}_{i,0:k-1}, \mathbf{z}_{1:k})}, \tag{7}$$

where q is an importance function and an element $w_{i,k}$ is ith element in the filter at time k. In many applications, it makes sense to choose q as the prior probability, in which case the weight update equation reduces to

$$w_{i,k} = w_{i,k-1}\mathbb{P}(\mathbf{z}_k|\mathbf{p}_{i,k}). \tag{8}$$

1. given a prior distribution $\mathbb{P}_0(\mathbf{x})$, let $w_{i,0} = 1/N$ and let each of $\mathbf{p}_{i,0} \sim \mathbb{P}_0(\mathbf{x})$ so that $P_0 = \{\mathbf{p}_{i,0}, w_{i,0} | i \in [1, N]\}$
2. $k = 1$
3. obtain the measurement $\mathbf{z}_k$
4. repeat the following steps for all particles
   (a) propagate through the dynamics f such that $\mathbf{q}_{i,k} = f(\mathbf{p}_{i,k-1}, \mathbf{v}_{i,k})$ where $\mathbf{v}_{i,k}$ is a value drawn from the true distribution as given by the model
   (b) update the weight according to $w_{i,k} = w_{i,k-1}\mathbb{P}(\mathbf{z}_k|\mathbf{q}_{i,k})$
5. increment k and go back to step 3

Knowledge of the measurement function h is used in the calculation of the probability $\mathbb{P}(\mathbf{z}_k|\mathbf{q}_{i,k})$.

Figure 1. The basic particle filter algorithm.

C. Two Problems and Methods for Detecting Them

The algorithm in Figure 1 works well, but there are two problems that can be encountered during implementation. Problem one is degeneracy, when the variance in the weights w_i is high. This results in a filter that has effectively fewer particles, since particles with weights far lower than average contribute vanishingly little information to knowledge of the distribution. Problem two is sample impoverishment, when most or all of the particles represent the same state ($\mathbf{p}_i = \mathbf{p}_j$). This reduces the amount of information available in the filter, since all particles with the same value will behave in the same way. Under either condition, filter performance degrades, where performance is measured by how closely the filter represents the underlying probability distribution, but computational effort remains the same.

Downloaded by Columbia University on November 3, 2026 | http://arc.aiaa.org | DOI: 10.2514/6.2005-6484

1. Number of Effective Particles

Degeneracy can be detected by calculating, or approximating, the variance in the weights. Referred to here as the effective number of particles, but introduced as the effective sample size,[15] it can be approximated as

$$N_{eff} \approx \left(\sum_{w_i \in w} w_i^2 \right)^{-1}. \tag{9}$$

When the weight of one particle is close to 1, the remaining particles in the filter hold no information (their probability of representing the true state is nearly zero). One goal in designing a particle filter is to keep all particles at approximately the same weight (if $w_i = w_j \ \forall \ i, j$ then $N_{eff} = N$) so as not to waste computation on useless particles.

2. Condensed Number of Effective Particles

Here, we introduce a new evaluation metric labeled N_{eff_c} for condensed number of effective particles. Consider the two cases of $\{\mathbf{p}_i, w_i \mid w_I = 1, w_{j \neq I} = 0\}$ and $\{\mathbf{p}_i, w_i \mid \mathbf{p}_i = \mathbf{p}_j \ \forall \ i, j\}$. For most cases, the two filters perform identically because identical particles are propagated and updated identically. We define N_{eff_c} as $N_{eff}(P^c)$ where P^c is the filter containing only one copy of each unique particle in P, with $w_i^c = \sum_{j \text{ s.t. } \mathbf{p}_i = \mathbf{p}_j} w_j$. [a]

This metric accounts for the number of different states in the distribution that are considered, and hence is a better approximation of the amount of information represented by the filter than N_{eff}. It detects the combined effects of sample impoverishment and degeneracy. To measure impoverishment alone, simply compare the number of particles in P^c to N.

D. Common Solutions

1. Resampling

Resampling is a technique used to combat degeneracy in particle filters. At intervals determined either statically[12] or dynamically,[16] the current set of particles is replaced by one drawn from the current estimate of the distribution according to Eq. (6). The weight on each new particle is set to $1/N$ and the update and propagation steps resume as usual. For this work, we used the residual sampling algorithm, which is outlined in Figure 2 and widely referenced in the literature.[17]

Resampling alone results in multiple copies of the same particle in the case where N_{eff} is low, and so another technique is needed to combat sample impoverishment.

Creating a new P_{k+1} using residual sampling.

1. copy $\text{floor}(N * w_{i,k})$ copies of particle $\mathbf{p}_{i,k}$ into P_{k+1}
2. let $w'_{i,k} = N * w_{i,k} - \text{floor}(N * w_{i,k})$ and $w' = [w'_{1,k} \dots w'_{N,k}]$
3. sample $\mathbf{p}_{i,k+1}$ from $P' = \{\mathbf{p}_{i,k}, \frac{w'_{i,k}}{\|w'\|}\}$ according to $\mathbb{P}(\mathbf{p}) = \sum_{i=1}^{N} w'_{i,k} \delta(\mathbf{p} - \mathbf{p}_{i,k})$ until P_{k+1} has N particles

Figure 2. The Residual Sampling algorithm[17]

2. Roughening

Roughening is used to reduce sample impoverishment. It is a method that creates new distinct particles to increase the amount of information represented in the filter. Instead of always resampling directly from the current particle set P_k for all of P_{k+1}, some or all of the particles have noise added to them. Roughening is used to prevent premature convergence, but can result in divergence if not done properly.

[a] In some applications, it may be better to test $\|\mathbf{p}_i - \mathbf{p}_j\| \ll \epsilon$, but this would require domain knowledge of the dynamics, as $\mathbf{p}_i$ and $\mathbf{p}_j$ could be diverging if, for instance, they are on opposite sides of a branch point.

Downloaded by Columbia University on November 3, 2026 | http://arc.aiaa.org | DOI: 10.2514/6.2005-6484

IV. A Modified Roughening Technique: Automatic Roughening

1. $P_{k+1} = \text{resample}(P_k)$ according to Figure 2
2. roughen all the particles in P_{k+1}:
   replace each $\mathbf{p} \in P_{k+1}$ with $\mathbf{q} = \mathbf{p} + \mathbf{c}$, where

$$\mathbf{c} \sim \mathcal{N}(0, J),\ J = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n)$$

   σ_j can be calculated using either:

Auto-Roughening			Roughening[12,16]		
σ_j	=	$\min_{\mathbf{p}\in Q} \lvert\mathbf{p}(j) - \mathbf{p}_i(j)\rvert$	σ_j	=	$GN^{1/n}E_j$
Q	=	$\{\mathbf{p} \mid \mathbf{p} \in P_{k+1}, \mathbf{p}(j) \neq \mathbf{p}_i(j)\}$	E_k	=	$\max_{\mathbf{p}_i,\mathbf{p}_l\in P_c} \lvert\mathbf{p}_i(j) - \mathbf{p}_l(j)\rvert$

   where $\mathbf{p}(l)$ is the lth element of $\mathbf{p}$ and G is a tuning factor that must be chosen by hand.
3. update the weights as the final step

$$w_{i,k+1} = \frac{1}{N}$$

Figure 3. Resampling a particle filter with roughening using the bootstrap method[12] and the proposed automatic roughening.

To increase the robustness of untended particle filters, we propose a roughening technique that automatically adjusts the noise level to prevent the filter from erroneously diverging or converging. The standard resampling and roughening step is outlined in Figure 3 along with our proposed modification. Our technique adds noise to new particles depending upon the local particle density. Automatic roughening uses this density information to keep the artificial noise small in high probability (high density) sections of the distribution and increase it in low probability sections to ensure the distribution is adequately covered.

Mathematically, it can be seen that our noise parameterization depends on nothing external to the particle filter. In addition, the original roughening method uses a globally applied noise factor while the proposed Auto-Roughening uses a factor tailored to each particle. We hypothesize that the extra computation of the noise factor will be balanced by a reduced need for resampling, but leave this for future study.

V. Experimental Comparisons

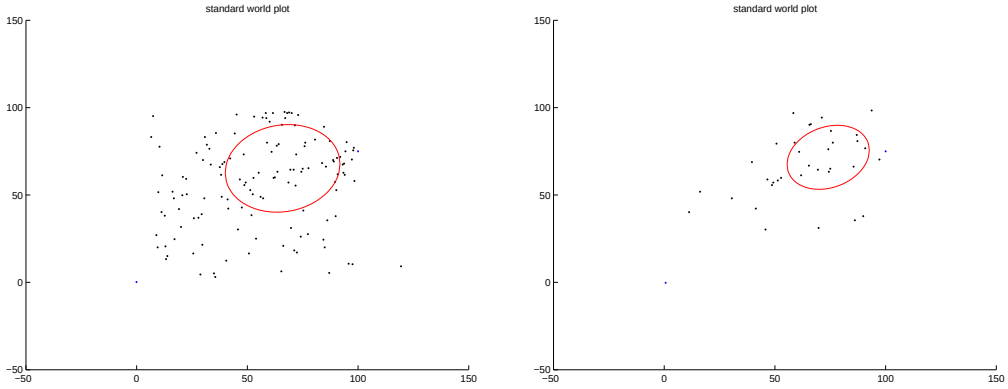
For this study, we used the dynamics models from Section II and placed them in a simplified problem. The vehicle is in an open environment that is known to contain one object. The goals of the exploration are twofold: obtain an accurate estimate of the possible locations of the object and approach the object to perform scientific experiments. Since the vehicle state is assumed known, the particle filter tracks on the the position of the object.

We have performed a number of experiments to demonstrate the utility of automatic roughening in the context of unmanned exploration. Averaged results are presented for performance evaluation, while individual cases are used to illustrate differences in operation.

Figures 4, 5, and 6 show the differences between the various resampling schemes. This is a sample run of a vehicle following a random walk near the point $(0, 0)$ while gathering information on an object located at $(100, 75)$ on the plot.

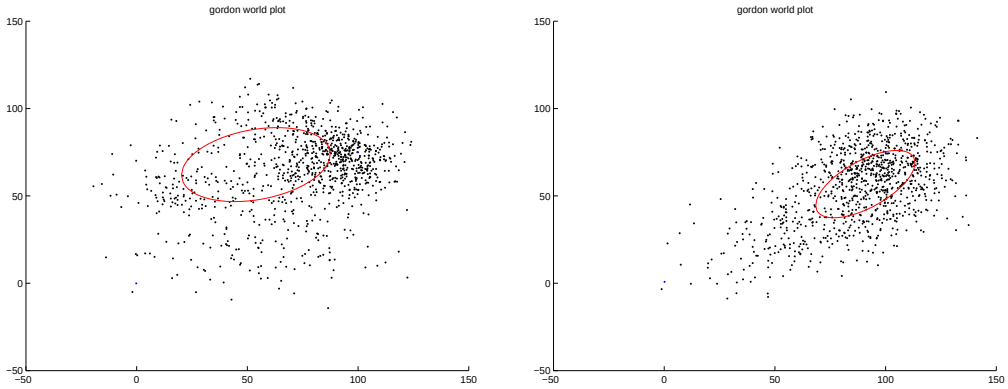
Figure 4 shows decent tracking performance, but demonstrates the sample impoverishment problem of the standard resampling technique. Figure 5 is the tuned resampling with roughening algorithm which maintains a much wider coverage of the distribution. The auto-roughening algorithm is shown in Figure 6 and maintains adequate coverage of the distribution while concentrating more of the particle weight along the axis of uncertainty (recall that the vehicle gets bearing measurements, but no range measurements).

Downloaded by Columbia University on November 3, 2026 | http://arc.aiaa.org | DOI: 10.2514/6.2005-6484



(a) Standard at t=5 (b) Standard at t=25

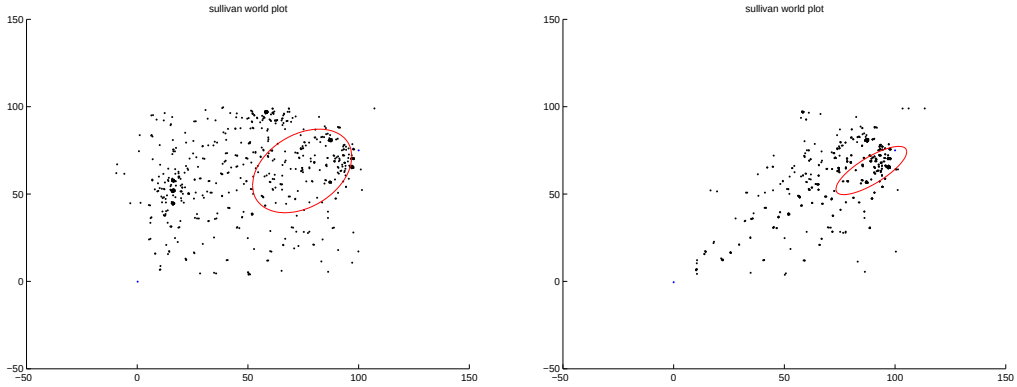
Figure 4. Storyboards showing the evolution of particles between times t=5 and t=25. The ellipse shows the mean-centered covariance of all the particles.



(a) Roughening at t=5 (b) Roughening at t=25

Figure 5. Storyboards showing the evolution of particles before and after the resampling step that occurred at time t=11. The ellipse shows the mean-centered covariance of all the particles.

Downloaded by Columbia University on November 3, 2026 | http://arc.aiaa.org | DOI: 10.2514/6.2005-6484



(a) Auto-Roughening at t=5

(b) Auto-Roughening at t=25

Figure 6. Storyboards showing the evolution of particles before and after the resampling step that occurred at time t=10. The ellipse shows the mean-centered covariance of all the particles.

Downloaded by Columbia University on November 3, 2026 | http://arc.aiaa.org | DOI: 10.2514/6.2005-6484

A. Experimental Setup

We present experimental results comparing three variations on particle filters. All of our Monte Carlo simulations use the same dynamics and begin with the same prior estimate. The first, called 'Standard,' is a bootstrap filter with update and resampling every step.[12] The second, called 'Roughened,' is the modified bootstrap filter with residual resampling and roughening when N_{eff_c} falls below $N/2$.[16] The Roughened filter has been hand tuned to work well on the sample problem. The third, called 'Auto-Roughened,' is the new technique presented here using the same residual resampling and N_{eff_c} threshold as in the Roughened version.

Each filter used a prior estimate, P_0, of 1000 particles uniformly distributed in the area $x \in [0, 100]$, $y \in [0, 100]$ with the vehicle starting at $(0, 0)$ and the stationary object located at $(100, 75)$. Relevant parameters were chosen to be $G = 2 \times 10^{-4}$ (Figure 3), $J = \frac{\pi^2}{16}$ (Eq. 2). Measurements and control were both updated every time step.

B. Root Mean-Square Error

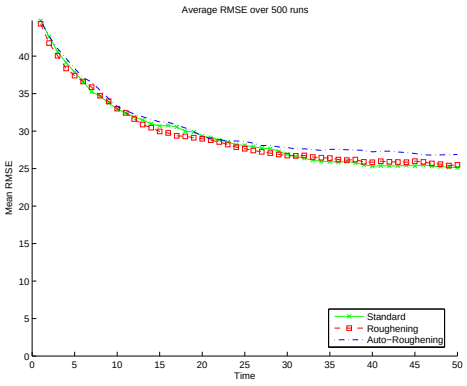
Root Mean-Square Error (RMSE, Eq. (10)) is currently the primary characteristic of concern for evaluating the performance for many types of filters. Figure 7(a) shows the error comparisons on a sample exploration problem. The results are similar for all three variants, which is as expected, since G was chosen with this problem in mind.

$$\mathrm{RMSE}(P, \mathbf{x}) = \sqrt{\sum_{i=1}^{N} w_i (\mathbf{p}_i - \mathbf{x})^2} \tag{10}$$

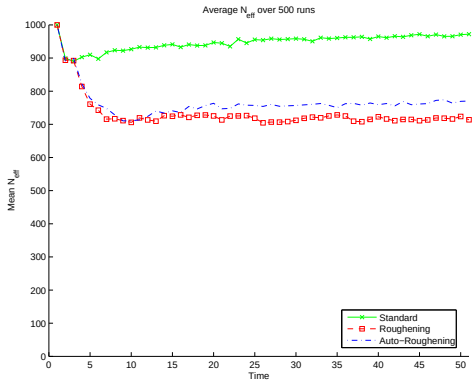
RMSE is not the best performance metric for particle filters, since the distributions they maintain can rarely be reduced to a mean value without loosing valuable information. In our particular case, however, we do eventually want all the particles concentrated around the true location of the object of interest, so it is adequate. When control algorithms are brought together with particle filters, we can measure the performance of the controller as a secondary measure of filter performance, provided we can distinguish control algorithm performance from the quality of the data fed to the algorithms.

C. Average N_{eff} and N_{eff_c}

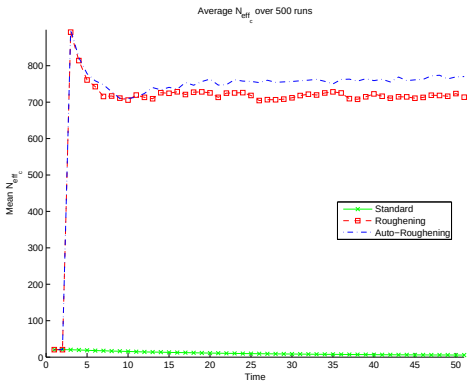
Our experiments show that roughened filters represent more information, on average, than the Standard filter when applied to the exploration problem. We present measures of N_{eff} and N_{eff_c} to demonstrate that this is the case and to show a drawback to using only N_{eff} to evaluate the particle filter's performance. Figure 7(b) shows results of measuring N_{eff} while Figure 7(c) presents data for N_{eff_c}. The Standard filter has a higher N_{eff} than either the Roughened or the Auto-Roughened. At first glance, this implies that the Standard filter maintains a better distribution of particles. Examining N_{eff_c} in Figure 7(c) shows that almost all the particles in the Standard filter



(a) RMSE averaged over 500 Monte Carlo runs



(b) N_{eff} over time averaged over 500 Monte Carlo runs



(c) N_{eff_c} over time averaged over 500 Monte Carlo runs

Figure 7. Collection of results for a sample 1 object exploration problem with the vehicle following a random walk

represent the same state. The Auto-Roughened filter, however, maintains a high N_{eff} and the highest N_{eff_c} of any of the filters. This means that the particles are more diverse in state-space, and the importance to modeling the distribution is spread evenly throughout the set of particles.

In addition, preliminary results show that the Roughened filter must resample up to four times more often than the Auto-Roughened filter. This reduction in computational load over traditional roughening is important, as resampling is a time-intensive process. It also implies that the particles chosen by Auto-Roughening are more appropriate, since the new particle set degenerates and becomes impoverished less quickly. We plan on studying the issue of intelligent resampling and its impact further.

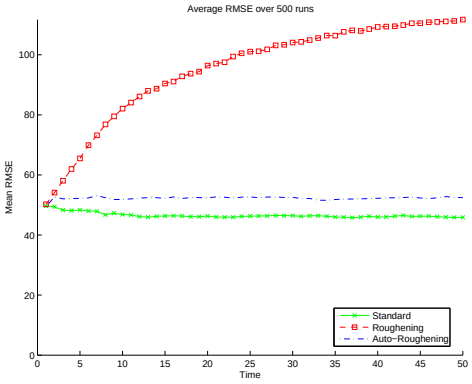
D. Changing the Model Parameters

Here, we present an experiment showing what can happen to a tuned filter when the model used in tuning is not accurate. This showcases the benefit of using an automatic approach, rather than one that requires exogenous parameters. In Figure 8, the sensor used in simulation was less accurate than assumed in the particle filter (Eq. 2 had $J = \frac{(10\pi)^2}{16}$ for simulation, but $J = \frac{\pi^2}{16}$ in the model). This simulates the sensor becoming broken after the rover has begun its mission, but the fault is undetected.

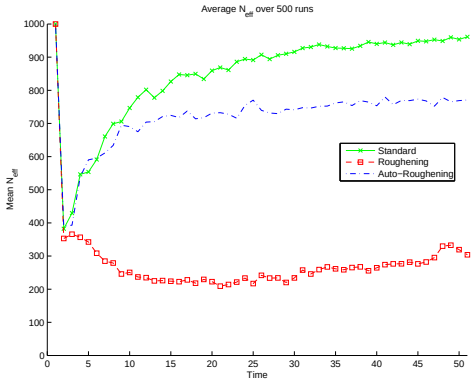
These results demonstrate the advantage in robustness to faults and model inaccuracies that come from using an adaptive Auto-Roughening filter instead of a traditional Roughening filter. The position estimate under Auto-Roughening does not get any better, but neither does it get any worse, as it does under the Roughening filter. The

Downloaded by Columbia University on November 3, 2026 | http://arc.aiaa.org | DOI: 10.2514/6.2005-6484

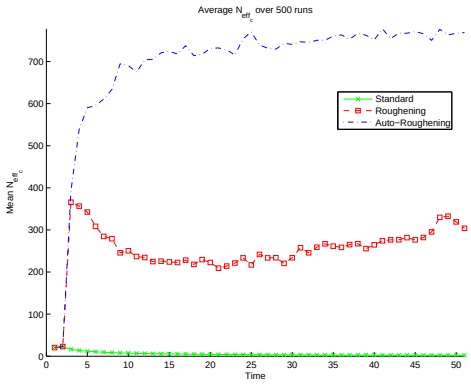
standard particle filter, without roughening, has similar RMSE performance to the Auto-Roughening filter in this context, but its sample impoverishment leaves it vulnerable to other problems.[12] Figure 8 implies that the Roughening filter prematurely converged on an incorrect location (falling N_{eff_c}, but high error), while the Auto-Roughening filter kept high coverage of possible areas (high N_{eff_c}) without commiting to a particular location (unchanging error over time).



(a) RMSE averaged over 500 Monte Carlo runs



(b) N_{eff} over time averaged over 500 Monte Carlo runs



(c) N_{eff_c} over time averaged over 500 Monte Carlo runs

Figure 8. Collection of results for a sample 1 object exploration problem after making the sensor *more* accurate than was assumed during tuning.

VI. Exploration Algorithms

The real goal of using a particle filter in an unmanned exploration context is not simply to estimate the state of the environment, but to do something with that information. Traditional filters output a mean and covariance upon which control strategies can make decisions. A particle filter, however, delivers a probability distribution, which requires a very different approach.

A. Greedy algorithm based on particle weights

Since the ideal particle filter has all particles with equal weighting, looking at weights alone will not provide much information. For examples such as this, however, a greedy algorithm is sufficient because the spatial distribution of the particles will narrow over time. If each particle is equally weighted, but also very close to the goal (the ideal case), then a greedy algorithm is all that is needed.

Downloaded by Columbia University on November 3, 2026 | http://arc.aiaa.org | DOI: 10.2514/6.2005-6484

Control inputs to Eq. 1 for the greedy algorithm are determined by

$$\begin{aligned} c_1 &= \begin{cases} 10 & v = 0 \\ 0 & \text{else} \end{cases} \\ c_2 &= \text{atan2}(\mathbf{p}_i(2) - y, \mathbf{p}_i(1) - x) - \theta, \text{ for } \max_i w_i. \end{aligned}$$

The distance over time between the object and a vehicle under a greedy algorithm is shown in Figure 9(b).

B. A simple information-gain algorithm

The complement to a simple greedy algorithm for goal-oriented behavior is a simple information-gain algorithm for exploration-oriented behavior. This algorithm was designed with a bearing-sensor in mind and moves the vehicle in the direction that will use the bearing sensor to 'triangulate' on the greatest number of particles to cull as many outliers as possible. This should increase accuracy, but sacrifice goal-seeking, as is shown in Figure 9(c).

Control inputs to Eq. 1 are determined by

$$\begin{aligned} c_1 &= \begin{cases} 1 & v = 0 \\ 0 & \text{else} \end{cases} \\ c_2 &= \psi + \frac{\pi}{2} - \theta, \text{ where } \begin{bmatrix} \cos\psi & \sin\psi \\ -\sin\psi & \cos\psi \end{bmatrix} \begin{bmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{bmatrix} \begin{bmatrix} \cos\psi & \sin\psi \\ -\sin\psi & \cos\psi \end{bmatrix}^T = \text{covariance}(\mathbf{p}_i). \end{aligned}$$

C. Mixture of exploration and goal-oriented behavior

A simple approach to combining exploration and goal-seeking is to use a mixture weight α. In this experiment, $\alpha = 0.9$ was used, meaning that 90% of the control action focused on heading towards the object using a greedy algorithm, while 10% of the effort focused on gaining better measurements. Results for this, shown in Figure 9(d), demonstrate slightly worse goal-seeking performance (as expected) as the trade-off for better information.

Control inputs to Eq. 1 are determined by

$$\mathbf{c} = \begin{cases} \eta < \alpha & \text{greedy control} \\ \eta \geq \alpha & \text{explore control} \end{cases}$$

where η is drawn from a uniform distribution on $[0, 1]$.

VII. Conclusions

A. Summary

We have proposed a new metric, condensed number of effective particles, and a new roughening method, Auto-Roughening, to move particle filters from tuned to adaptive methods. We showed that Auto-Roughening is comparable to traditional roughening under normal circumstances, but that it has a higher robustness when the model used in tuning is not an accurate representation of the world (which is too often the case).

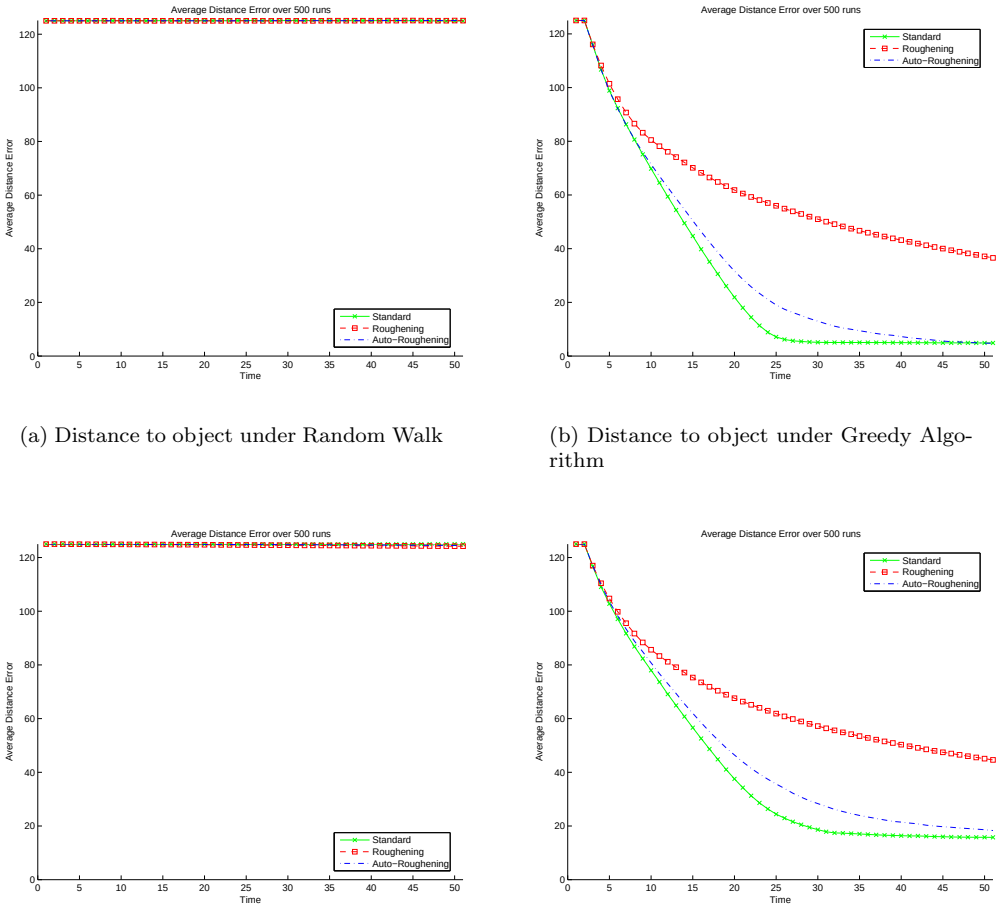
In addition, we have begun an effort to develop algorithms that work with particle filters to not only maintain an accurate state estimate, but use that estimate in its probability-distribution form. We present a few naive algorithms and their resulting performance on a single-object location and approach task.

B. Future Efforts

Our efforts in the future will concentrate on two main areas. The first is improving the intelligence of resampling and roughening steps to reduce the computational load required to represent large state spaces. We used a simple problem here for demonstration purposes, but are interested in applying these techniques to much larger systems.

The second area is that of control using particle filters. Using the mean and covariance of the particles to drive a controller is throwing away one of the largest benefits to particle filtering: the maintenance of non-Gaussian probability distributions over the true state. As mentioned in the introduction, a number of techniques can be used if increased information is the main goal, but little work has yet been done on incorporating other tasks into the control.

Downloaded by Columbia University on November 3, 2026 | http://arc.aiaa.org | DOI: 10.2514/6.2005-6484



(a) Distance to object under Random Walk

(b) Distance to object under Greedy Algorithm

(c) Distance to object under Exploration Algorithm

(d) Distance to object under Mixture Algorithm ($\alpha = 0.9$)

Figure 9. Collection of results for a sample 1 object exploration problem under different control schemes

Downloaded by Columbia University on November 3, 2026 | http://arc.aiaa.org | DOI: 10.2514/6.2005-6484

Acknowledgments

This work was supported in part by a NASA Graduate Student Research Program Fellowship number NNA04CK53H through the Evolvable Systems group at NASA's Ames Research Center.

References

[1]Schultz, A. C., Adams, W., and Yamauchi, B., "Integrating Exploration, Localization, Navigation and Planning with a Common Representation," *Autonomous Robots*, Vol. 6, No. 3, 1999, pp. 293–308.

[2]Lu, F. and Milios, E., "Globally consistent range scan alignment for environment mapping," *Autonomous Robots*, Vol. 4, No. 4, Oct. 1997, pp. 333–349.

[3]Lisien, B., Morales, D., Silver, D., Kantor, G., Rekleitis, I., and Choset, H., "Hierarchical Simultaneous Localization and Mapping," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vol. 1, October 2003, pp. 448–453.

[4]Welch, G. and Bishop, G., "An Introduction to the Kalman Filter," Tech. Rep. TR 95-041, Department of Computer Science, University of North Carolina, Chapel Hill, North Carolina, April 2004.

[5]Grocholsky, B., Makarenko, A., and Durrant-Whyte, H., "Information-Theoretic Coordinated Control of Multiple Sensor Platforms," International Conference on Robotics and Automation, 2003.

[6]Oh, S., Russell, S., , and Sastry, S., "Markov Chain Monte Carlo Data Association for General Multiple-Target Tracking Problems," *Proceedings of the 43rd IEEE Conference on Decision and Control*, December 2004, pp. 735–742.

[7]Rosencrantz, M., Gordon, G., and Thrun, S., "Locating Moving Entities in Indoor Environments with Teams of Mobile Robots,"

Proceedings of Second International Conference on Autonomous Agents & Multi Agent Systems, AAMAS'03, Melbourne, Australia, July 2003.

[8]Kreucher, C., III, A. O. H., Kastella, K., and Chang, D., "Efficient Methods of Non-myopic Sensor Management for Multitarget Tracking," *Proceedings of the 43rd IEEE Conference on Decision and Control*, December 2004, pp. 722–727.

[9]Ousingsawat, J. and Campbell, M. E., "Establishing Trajectories for Multi-Vehicle Reconnaissance," *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, Providence, Rhode Island, August 2004.

[10]Kreucher, C., Kastella, K., and III, A. O. H., "A Bayesian method for integrated multitarget tracking and sensor management," *Proceedings of the Sixth International Conference of Information Fusion*, 2003.

[11]Hue, C., Le Cadre, J.-P., and Perez, P., "Tracking Multiple Objects with Particle Filtering," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 38, No. 3, July 2002.

[12]Gordon, N., Salmond, D., and Smith, A., "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," *IEEE Proceedings on Radar and Signal Processing*, Vol. 140, No. 2, 1993, pp. 107–113.

[13]Arulampalam, M. S., Maskell, S., Gordon, N., and Clapp, T., "A Tutorial on Particle Filters for On-line Non-linear/Non-Gaussian Bayesian Tracking," *IEEE Transactions on Signal Processing*, Vol. 50, No. 2, February 2002, pp. 174–188.

[14]Julier, S., Uhlmann, J., and Durrant-Whyte, H. F., "A New Method for the Nonlinear Transformation of Means and Covariances in Filters and Estimators," *IEEE Transactions on Automatic Control*, Vol. 45, No. 3, 2000.

[15]Kong, A., Liu, J. S., and Wond, W. H., "Sequential Imputations and Bayesian Missing Data Problems," *Journal of the American Statistical Association*, Vol. 89, No. 425, March 1994, pp. 278–288.

[16]Cheng, Y. and Crassidis, J. L., "Particle Filtering for Sequential Spacecraft Attitude Estimation," *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, August 2004.

[17]Beadle, E. R. and Djuric, P. M., "A Fast Weighted Bayesian Boostrap Filter for Nonlinear Model State Estimation," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 33, No. 1, January 1997, pp. 338–343.

Downloaded by Columbia University on November 3, 2026 | http://arc.aiaa.org | DOI: 10.2514/6.2005-6484