Self-Driving Microscopy: Bayesian Inference for Localization and Autonomous Navigation

Eric Wadkins, Michael Walsh, Dirk Englund
{ewadkins, mpwalsh, englund}@mit.edu

Massachusetts Institute of Technology
Research Laboratory of Electronics
Quantum Photonics Group

***Abstract*—Self-driving microscopy is a promising method of increasing the efficiency and precision of laboratory experiments, while also decreasing the cost of laboratory operations. However, precise instrument localization is a difficult task due to error generated by physical instruments. Techniques such as Bayesian inference allow for a noise-tolerant localization process, but there are currently few implementations for applications of microscopy, which often involve operating in an environment with the potential for extreme observational error. Here, I propose the design and implementation of a self-driving microscopy system that is capable of localization and autonomous navigation given only partial, and potentially imperfect, information. This system uses the approach of Monte Carlo Localization (MCL) and various improvements to the basic MCL implementation to ensure efficient belief convergence. The aim of this work is to provide the Quantum Photonics Group at MIT's Research Laboratory of Electronics with a self-driving microscopy setup, allowing experiments to be automated.**

I. Introduction

The most challenging aspect of the creation of any self-driving system is its ability to understand its location within the surrounding world — a task known as localization. For applications that must track and make decisions based on the location of mobile instruments, localization is of critical importance. One difficulty, however, is that physical systems are commonly subject to some amount of noise as a result of operational error. If left unchecked, this error will amplify over time, rendering the physical system unable to complete its task.

Like many applications, microscopy stands to benefit from automation, but also suffers from the challenges inherent to such an automated system. A self-driving system must be able to perform accurate localization. However, at the extreme scales that microscopy systems generally operate at, instrument error can have a profound impact on the accuracy and overall performance of the localization process. Worse still, environmental noise, such as a piece of dust or a fabrication error, can make localization even more challenging. Similar problems are commonplace in many fields, but more prevalent in applications of microscopy. This is likely part of the reason few automated microscopy systems currently exist.

The Quantum Photonics Group at MIT's Research Laboratory of Electronics uses microscopy to view nitrogen-vacancy centers in diamond, which appear only on the scale of microns. An automated system capable of global localization and autonomous navigation would be useful for automating laboratory experiments, which would, in turn, decrease time and cost spent on each experiment. In this paper, I detail the design of a self-driving microscopy system that is capable of localization and autonomous navigation given only partial, and potentially imperfect, location information. In order to ensure resilience to error, the system uses Monte Carlo Localization (MCL) with various improvements to the base implementation to increase the efficiency and performance of the system.

II. Related Work

Many approaches currently exist that are capable of achieving global localization, or the task of locating oneself within the world given an initial uniform belief distribution. Most implementations use a type of Markov localization, which maintains a belief distribution over all possible hypotheses as observations are made. These hypotheses are grouped by filters to make the task of maintaining all hypotheses more manageable. For example, the simplest approach of Markov localization uses a histogram as a filter, which can be thought of as discrete grid. Each cell has a certain probability of

the instrument being located there. The problem with this approach is that the precision of the algorithm is inversely proportional to the performance because the resolution of the grid is the determining factor for both. This is a serious problem when performance and precision are both of paramount importance, as would be the case for a precision real-time self-driving system.

Another common approach that strikes a balance between performance and precision is known as Monte Carlo localization (MCL), a Markov localization implementation with particle filters, rather than a histogram filter [1]. Each particle represents a location of interest. They are randomly distributed over the domain for the first iteration, assuming no information regarding the instrument's location is known initially. At each iteration the particles are weighted according to the likelihood that the instrument is located there. In subsequent iterations, particles are resampled based on the weights of the previous iteration's particle set. So, over time the particles cluster around the most likely location, as seen in Fig. 1. Because the particles are points in a continuous space, the precision using this approach is as high as the observations allow for. The Quantum Photonics Group will need precision down to roughly 10 nanometers, making the MCL approach much better-suited than the grid-based approach.

The proposed system will also provide active localization, which is the process of navigating the instrument in the direction of maximum expected information gain in order to achieve rapid belief convergence. A few active implementations exist, such as the one by Jensfelt and Kristensen [2]. Jensfelt et al.'s implementation is heavily dependent on heuristics that use specific features of the environment, such as wall boundaries. Not only are these heuristics not guaranteed to result in the optimal convergence time, but they can occasionally result in increased distance from the goal position. The implementation could be modified to decrease the time it takes to navigate to the goal position as well as the convergence time. Also, while these previous implementations offer inspiration, ideally the proposed system would be generalizable to any setting. I've also drawn inspiration from Engelson et al. [4] when taking error into consideration in the active localization process.

Many of the systems mentioned above include desirable features for a self-driving microscopy system — namely the use of Monte Carlo Localization for global localization, as well as the potential for active localization. However, improvements must be made to these approaches for use in a system that is generalizable and error-resistant enough to function in any application of microscopy. These approaches serve as inspiration, as well as provide a baseline implementation to expand upon and use in the evaluation of the proposed system.

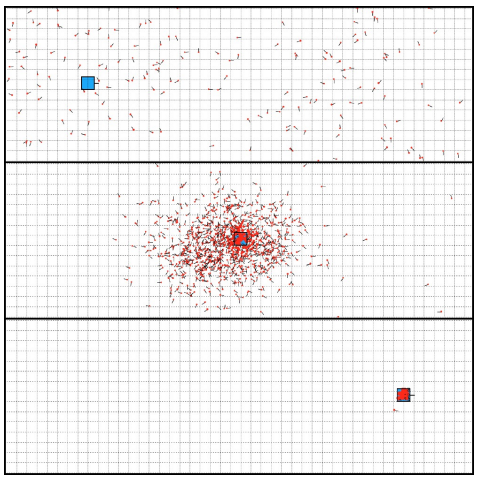

Fig. 1. **Using MCL, the belief distribution is updated as an instrument moves across an environment and makes observations.** The red dots represent the particle filters used by the MCL algorithm to converge to the actual location. Collectively, they represent the belief distribution.

III. Methods

A. Primary Tasks

The system laid out in this paper accomplishes three primary tasks: global localization, local localization, and autonomous navigation of laboratory instruments. Global localization is necessary in order for the system to discover the location of an instrument with no prior knowledge of its location. The Monte Carlo Localization algorithm provides an efficient method of achieving global localization as well as local localization — the task of maintaining a correct belief during system operation and instrument motion.

Local localization through this method alone, however, is not a sufficient, as the system must be resilient to observational and environmental error and have the ability to rapidly recover from erroneous observations. This is achieved through modifications to the MCL algorithm. Specifically, the redistribution rate of particles is lowered to ensure that only multiple consecutive erroneous observations could result in the system holding an incorrect belief. This parameter has been set to a predefined value determined by empirical observations.

Autonomous navigation is the final task the system must be capable of achieving. For the system to be beneficial in laboratory environments, it is critical that this task be completed in an efficient manner. For this reason, the system utilizes active localization in situations where the belief confidence is less than some predefined threshold.

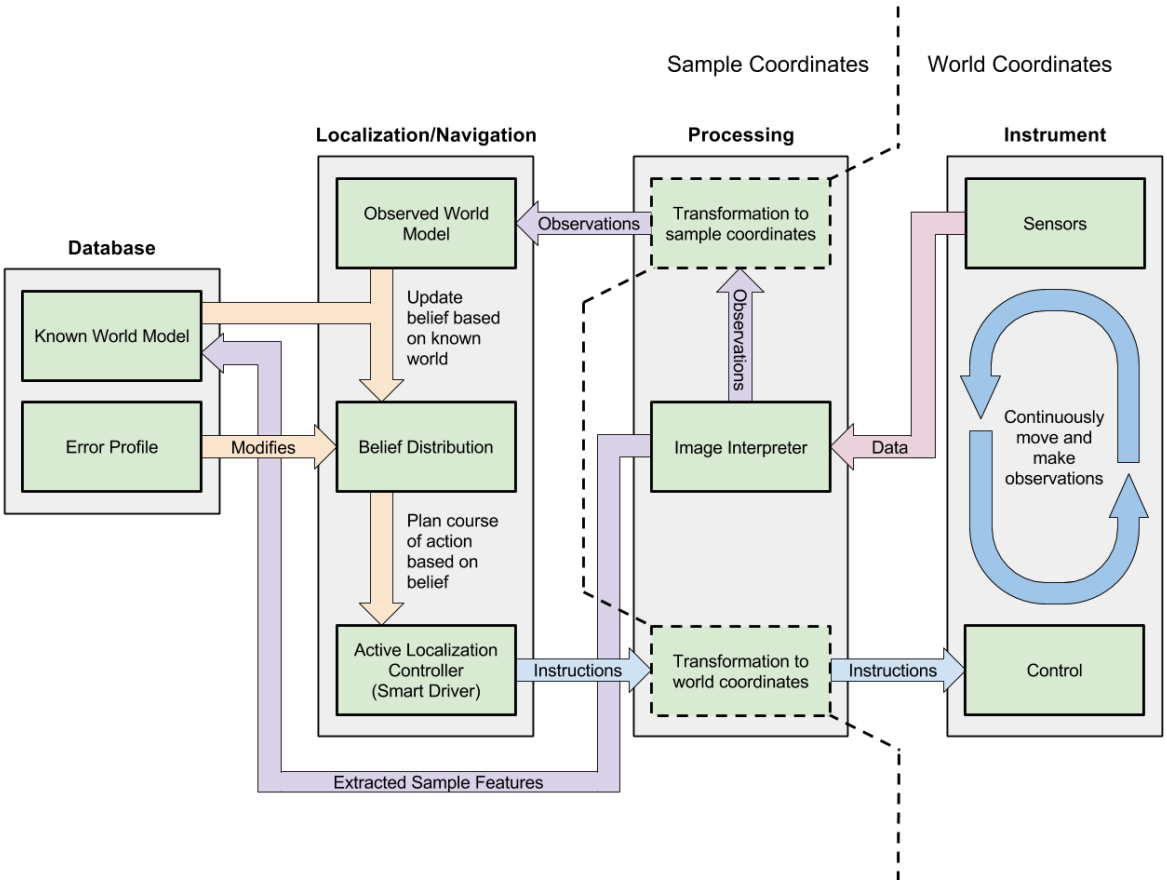

Fig. 2. **A system overview detailing the interactions between the instrument, the localization and navigation system, and the database in which the known world model and error profile are stored.** Notice the feedback loop that originates with the instrument sensors and concludes with the instrument control through the observations and instructions pipeline in the localization/navigation module.

Fig. 2 shows an overview of how the system components interact to accomplish these tasks. Further details are available in the Technical Approach section.

B. Analytical Methods

Implementations of the three primary tasks are evaluated through the use of a modular system, capable of testing via a simulation or physical laboratory hardware. The module design also serves to simplify the process of integration with laboratory hardware. Several variations of the system, each with a unique feature, have been tested on identical tasks in order to test the performance and significance of each feature in isolation.

The simulation exactly mimics a physical instrument in its use of the system, providing accurate data that can be used in the analysis of the system. The state of the simulation is preserved after the completion of an experiment so that the state history can be analyzed and success metrics can be calculated. These success metrics include belief convergence time, ratio of realized path length to optimal path length, and deviation from optimal path after belief convergence — all of which are directly calculable using only data on the instrument's path, belief over time, the goal position, and the instrument's initial position.

C. System Architecture

The command center that controls the laboratory hardware of the Quantum Photonics Group is written in MATLAB. The system has been designed as a collection of Python modules that are able to be referenced from a MATLAB environment. The simulation has been implemented in Python and interacts with the modules in an identical manner. NumPy and SciKit were used in the development of the MCL algorithm and driver, and Tkinter is used in the simulation's graphical interface. An example view of the simulation can be seen in Fig. 3.

IV. Technical Approach

For the reasons outlined in the Related Works section, my technical approach focuses on an implementation of Monte Carlo Localization (MCL) and various

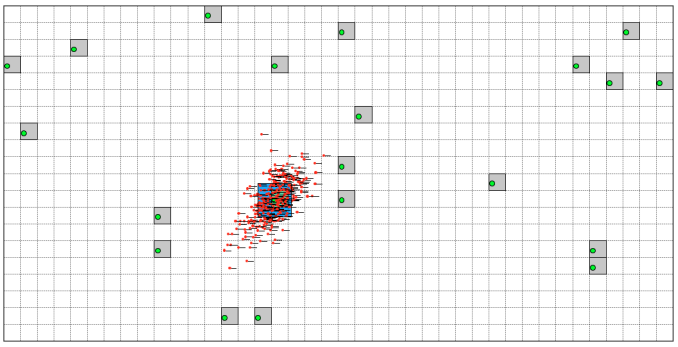

Fig. 3. **A screenshot of the simulation used in evaluation of system features.** The instrument is shown in blue, and the hypotheses of the instrument's locations, given the instrument's current field of view, is represented by the green dots. The red particles collectively represent the belief distribution. Over time, the particles converge to the instrument's actual location, a process which is shown here in progress.

improvements. MCL provides an efficient, arbitrarily precise solution to the problem of global localization. The following improvements further improve the implementation, making it more suitable to applications of microscopy.

A. *Resampling Rate*

The resampling rate of MCL, which is the percentage of the particles that are resampled during each update iteration, has a large effect on the belief convergence process. More specifically, it allows for a tradeoff between rapid convergence and error resilience. This is because if the resampling rate is very high, the system is able to learn more quickly its location *given correct observations*. However, at the same time, higher resampling rates make the system less resilient to erroneous observations, since each observation has a larger effect on the current belief when the resampling rate is high. On the other hand, if the resampling rate is too low, less confidence is placed on each observations, meaning localization may take longer. However, the lower the resampling rate is, the more resilient to erroneous observations the system is.

This tradeoff between convergence time and error resilience is crucial and depends heavily on the environment (and the amount of error present in the environment). Fig. 4 shows the resampling rate vs. the ratio of realized/optimal path lengths taken by the simulated instrument — a lower value is better. The simulation uses an environment with 20% error, which is quite high; however, the results look similar for a large range of error rates. As you can see in Fig. 4, a resampling rate that is high, but less than one, is ideal. In other words, resampling/repositioning the majority of the particles, but not quite all of them, is beneficial. This makes intuitive sense when one considers the fact that making two or more erroneous observations consecutively is unlikely. So, even if only a few particles remain in the correct region after an erroneous observation, the next observation would correct this error and reposition the particles to the correct region once again. This approach solves one root cause of what is commonly known as the particle-deprivation problem, which is erroneous observations, while also allowing for rapid belief convergence.

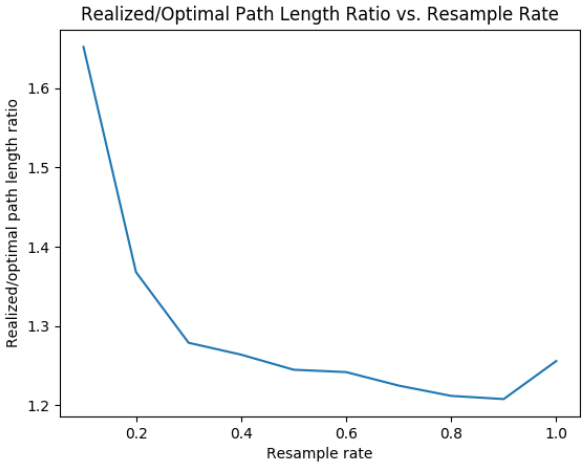

Fig. 4. **A high resample rate, but one that is less than 1, is ideal in environments with error.** This figure shows the resampling rate vs. the ratio of realized/optimal path lengths (so a lower y-value indicates better performance).

B. *Redistribution Rate*

Using ideas from Thrun et al. [3] regarding sampling techniques, I added the concept of a redistribution rate to the implementation of the MCL algorithm. It works as follows: after each particle resampling, a small subset of the particles, chosen according to the redistribution rate, are randomly distributed over the environment. This decreases the probability that any region of the environment will be entirely devoid of particles — which in turn means that it is possible for the system to conclude that any region of the environment is the correct location of the instrument. Without redistribution in this manner, consecutive erroneous observations could cause the correct region of the environment to become devoid of particles (the particle-deprivation problem), leaving the system unable to recover the correct belief. The redistribution rate is particularly important in the early stages of the localization process, but becomes less important as belief convergence is achieved — after

which the resampling rate becomes more important due to its contribution to error resilience.

Fig. 5 shows the realized/optimal path length ratio vs. the redistribution rate in a testing environment with few particles. An environment with few particles was chosen for testing because in such an environment particle deprivation becomes a risk. To protect against particle deprivation, as explained above, the redistribution rate can be increased to some non-zero value. As you can see in the figure, a low but non-zero redistribution rate clearly gives the best performance. This is because, similarly to the resampling rate, the redistribution rate allows for a tradeoff — one between convergence time and the risk of particle deprivation. If the redistribution rate is too high, the belief convergence process takes longer, as the system is not able to learn as quickly. However, as explained above, if the redistribution rate is too low, particle deprivation, which leads to drastically increased convergence times, becomes a risk.

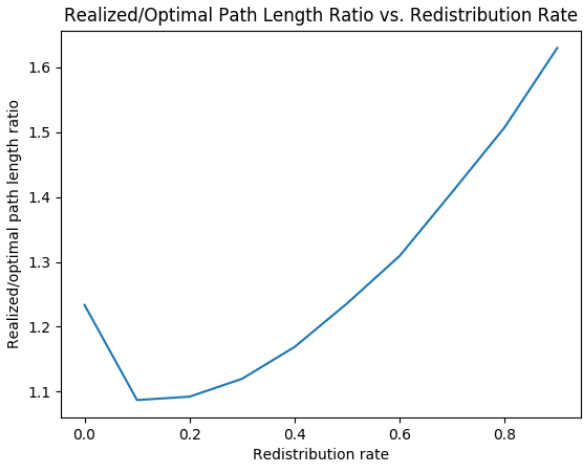

Fig. 5. **A low but non-zero redistribution rate is ideal in environments with few particles.** This figure shows the redistribution rate vs. the ratio of realized/optimal path lengths (so a lower y-value indicates better performance).

C. KLD Sampling

Due to the nature of the MCL algorithm, the more particles that are used, the faster the system is able to converge to the correct belief. However, more particles also increases the computation time of each update step. Since the system being developed for applications of microscopy must run in real-time, the particle count is therefore a large concern. It is important to note that as convergence progress, less and less particles are necessary (or helpful, for that matter). Therefore, ideally the system would respond by reducing the particle count according to how close to convergence the system is.

KLD sampling, in the context of the MCL algorithm, is the process of updating the number of particles based on necessity. The variance of the set of particles is used to determine an updated particle count. When there is more confidence in the belief, indicated by a low variance in the particle set, then the number of particles is reduced to save computation time. This process occurs according to the following formula, where n_0, the initial number of particles, is determined empirically, n_{min} is the minimum number of particles allowed, σ^2 is the variance of the particle set, and α is an arbitrary scalar applied to the variance:

$$n = \max(n_{min}, \lceil n_0 \cdot \min(\sigma^2/\alpha, 1) \rceil); \qquad (1)$$

V. Results

We've already discussed the benefits of using the tuned resampling rate, redistribution rate, and KLD sampling to improve performance. Here, I will discuss the simulated MCL implementation with these various improvements included, as described in the Analytical Methods section. The implementation of MCL described above, after tuning to the specific environment, was able to achieve a ratio of realized/optimal path lengths of 1.23, given the task of reaching a goal position 800 microns away. This means that while the optimal path was 800 microns in length, the path taken by the instrument was $1.23 \cdot 800 = 984$ microns — a difference of 184 microns. Given that each data point is separated by about 40 microns, this means the instrument, on average, had to move enough to observe 4 data points before gaining a strong belief as to where it was located relative to the goal position. This is a promising result considering each observation in this environment contained only 4 bits of information. Given a higher information density, the ratio decreases further — achieving a ratio of about 1.06 given 6 bits of information in each observation. Fig. 6 shows the paths taken by the simulated instrument through the two aforementioned environments. You'll notice that the path is shorter for the 6-bit environment, which indicates that the system was able to converge to the correct belief more quickly. This difference in realized path length across environments shows the system's ability to learn about its environment using only observations of it. Also notice the small irregularities in the path once the system has achieved convergence (the relatively straight line segment). These are a result of erroneous observations in the environment. The system is able to handle these incorrect observations without its belief being drastically altered, and therefore it continued traveling along the

most direct path to the goal position. (Note: I still need to include details about how the improvements made (see Technical Approach) affected the accuracy and performance of the system. This data has not been collected yet).

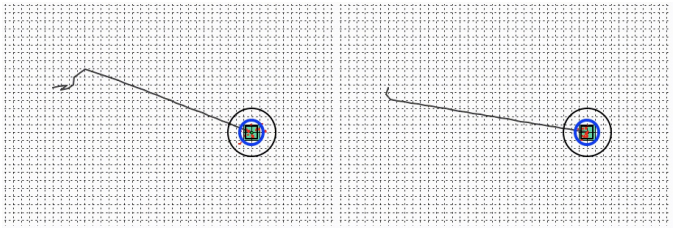

Fig. 6. **The system has a clear advantage in environments with higher information densities — illustrating the significance of learning through observations.** On the left is an environment with 4 bits of information per observation, and on the right is a 6-bit environment. Notice the difference in realized paths between the two environments. The 6-bit environment shows a clear advantage in being able to reach the goal position more quickly, and the system is able to take this extra information into account using MCL.

The active localization implementation failed to achieve any meaningful improvement over the implementation that utilized only the tuned resampling rate, redistribution rate, and KLD sampling features. In fact, during many trials, it led to worse performance. Results show that the implementation without active localization was able to achieve a realized/optimal path length ratio of 1.06. The implementation with active localization was only capable of achieving around 1.27 — meaning that on average, the active localization introduced an additional 21% error into the localization and navigation system. With that said, in roughly 12% of the trials, the active implementation was able to achieve better performance than the baseline.

This could be the result of noise, or it could be indicative of a more challenging problem that would need to be addressed to improve the performance of the active localization. My hypothesis as to why, on average, the active localization worsens the accuracy but is sometimes able to improve it is that the space of subsequent actions that would need to be explored in order to implement the optimal active localization algorithm is continuous. This is because the instrument could potentially move any distance (less than the instrument's step size) in any direction. Each one of these many actions would result in a different position of the instrument and a different observation. For computational reasons, only a small portion of this action space can be explored, as this exploration process occurs for each of the hypothesis, and performing an exhaustive search would prevent the system from operating in real-time (such an implementation was created, and the runtime of each iteration increased from on the order of tens of milliseconds to on the order of tens of seconds). Because of this reduction in the action space, it is quite likely that at each iteration of the MCL algorithm the optimal action to take is not being explored, and therefore it may be better in most scenarios to ignore the active localization heuristic entirely and simply move in the expected direction of the goal position, as the base implementation does. For this reason, active localization was not used during testing of the other features and the system as a whole, as reported previously.

VI. Conclusion and Discussion

The system succeeds in automating laboratory tasks in an accurate, precise, and error-resilient manner. The improvements have shown considerable improvement over the baseline MCL implementation. This successful automation of laboratory task illustrates two things: 1) MCL and the improvements made is suitable for the task of global localization and autonomous navigation in the context of applications of microscopy, and 2) inexpensive laboratory equipment, such as the ones modeled in this paper, can be used in lieu of more expensive hardware the is capable of achieving the same task through closed-loop control systems. So, not only does this "smarter" system save time regarding the completion of laboratory experiments, but it also has the potential to save laboratories considerable amounts of money that would otherwise be spent on hardware.

Still, there remains more work to be done. First, the active localization implementation shows little to no advantage over the system without that active localization (sometimes even worsening the accuracy). This is likely a result of the environment used in testing. Nonetheless, this feature requires more testing. Second, the system should be tested on different data-encoding schemes. The ones discussed in this paper all used a randomly generated environment containing k bits in each observation. This is likely not the case in many settings, however, and it is not unreasonable to expect the accuracy of the system to fall in other more organized environments. Finally, the system needs to be evaluated once it is fully integrated into laboratory hardware. Currently, the simulation has been the primary mode of testing. While the simulation was modeled after real environments, complete laboratory integration is the next logical step in the evaluation process. These next steps should prove the worth in using such a "smart" self-driving microscopy system in a laboratory setting.

VII. Acknowledgements

The work presented in this paper was supported by MITRE and the MIT Department of Electrical Engineering and Computer Science through an Undergraduate Research and Innovation Scholarship.

References

[1] Fox, D., Burgard, W., Dellaert, F., Thrun, S. "Monte Carlo Localization: Efficient Position Estimation for Mobile Robots." ACM, AAAI, 1999.

[2] Jensfelt, P., Kristensen, S. "Active Global Localization for a Mobile Robot Using Multiple Hypothesis Tracking." IEEE, Transactions on Robotics and Automation, Volume 17, Issue 5, 2001.

[3] Thrun, S., Fox, D., Burgard, W. "Monte Carlo Localization With Mixture Proposal Distribution" AAAI, 2000.

[4] Engelson, S., McDermott, D. Error correction in mobile robot map learning. ICRA, 1992.