# Effects of Obstacle Avoidance to LQG-based Motion Planners


## Abstract

Motion planning under uncertainty is critical for robust autonomy. The Partially Observable Markov Decision Process (POMDP) is a principled and general framework for solving such problems. Although solving a POMDP problem exactly is computationally intractable, in the past decade, many practical methods have been proposed to approximate solution of POMDPs that represent motion planning under uncertainty problems. However, the problem remains relatively open when it involves robots with complex non-linear dynamics. Recently, linearization-based methods that are derived from the Linear Quadratic Gaussian (LQG) controller have been shown to perform well in some planning under uncertainty problems with non-linear robot dynamics. However, it is not clear what the effect of linearization to motion planning under uncertainty is. The control and estimation results have clearly indicated that linearization performs well only when the non-linear dynamics is "weak". These results will definitely apply to the LQG-based derived methods for solving motion planning under uncertainty problems, too. However, a significant difference between motion planning and control is in the presence of obstacles. And it is not clear how the presence of obstacles affect the effectiveness of linearization and Gaussian simplification for solving motion planning problems. This paper presents a preliminary study in understanding this effect, by performing comparison studies via simulation.


## 1 Introduction

An autonomous robot must devise reliable motion strategies that avoid collision with obstacles and satisfy its kinematics and dynamics constraints, despite not knowing the exact effect of its actions, despite various errors in sensing and perception, and despite unpredictability of the environment. A systematic and principled approach to automatically construct such a strategy is to frame the problem as a stochastic motion planning problem —quantifying uncertainty using probability distribution function and computing the best strategy taking into account this quantified uncertainty. One of the base framework for this approach is the Partially Observable Markov Decision Process (POMDP) [12; 16]. POMDP represents the aforementioned types of uncertainty as probability distributions, and estimates the system's states as a probability distributions called *beliefs*. It computes the best strategy with respect to beliefs rather than with respect to single states, because the actual state is uncertain due to errors in the system's dynamics and sensing. Although the concept of POMDP was proposed in early 1960 [8], only in recent years that it starts to become practical for robotics problems (e.g., [11; 22]).

We can classify existing practical POMDP solvers into two approaches. First is the general POMDP solvers. These solvers do not restrict the type of dynamics and sensing model of the system, nor the type of distributions used to represent uncertainty. It can now compute good motion strategies on-line with 1-10Hz update rate for a number of robotics problems [13; 19; 20; 18]. However, their speed degrades in problems with complex non-linear dynamics, such as problems where unknown friction properties must be taken into account.

The second approach linearizes the system's dynamics model and restricts the distribution to always be Gaussian [21; 1; 5; 6; 17]. These methods are often derived from stochastic control methods, such as the LQG (Linear Quadratic Gaussian) controller [7]. It does promise better speed in handling complex non-linear dynamics, as it simplifies the dynamics model and takes into account only the mean and variance of the distribution. However, it is not clear when and where would this strategy works well for motion planning under uncertainty.

The control and estimation results have clearly indicated that such linearization performs well only under certain properties of the system, namely when the non-linear dynamics is "weak" [15]. These results will definitely apply to the LQG-

based derived methods for solving motion planning under uncertainty problems, too. However, a significant difference between motion planning and control is in the presence of obstacles. And it is not clear how the presence of obstacles affect the effectiveness of linearization and Gaussian simplification for solving motion planning problems. This paper presents a preliminary study in understanding this effect, by performing comparison studies via simulation.

## 2 Background and Related Work

### 2.1 POMDP Background

A POMDP problem is defined as a tuple $\langle S, A, O, T, Z, R, b_0, \gamma \rangle$, where $S$, $A$ and $O$ are the robot's state, action and observation spaces. $T$ denotes a a conditional probability function $p(s'|s,a)$, where $s, s' \in S$ and $a \in A$, that models the uncertainty in the effect of applying an action, when the robot is in a particular state. $Z$ denotes the conditional probability function $p(o|s,a)$ that models the uncertainty in the observations the robot receives. $R : S \times A \mapsto \mathbb{R}$ is a state-action dependent reward function, $b_0$ is an initial belief, and $\gamma \in (0,1)$ a discount factor that balances immediate rewards against future rewards.

At any given time, the POMDP agent (representing the robot) is at a state $s \in S$. It takes an action $a \in A$, perceives an observation $o \in O$, and moves to next state in a single time step. As a side effect of this movement, the agent receives a reward based on the reward function $R(s,a)$, and moves to the next state. Due to uncertainty in the results of action and sensing, the robot never knows its exact state and therefore, estimates its state as a probability distribution, called belief. The solution to the POMDP problem is an optimal policy (denoted as $\pi^*$), which is a mapping $\pi^* : \mathbb{B} \rightarrow A$ from beliefs ($\mathbb{B}$ denotes the set of all beliefs, which is called the belief space) to actions that maximizes the expected total reward the robot receives, i.e., $V^*(b_0) = \max_{a \in A} \left( R(b,a) + \gamma \int_{o \in O} p(o|b,a) V^*(\tau(b,a,o))\, do \right)$, where $\tau(b,a,o)$ computes the updated belief estimate after the robot performs action $a \in A$ and perceived $o \in O$ from belief $b$, and is defined as:

$$b'(s) = \tau(b,a,o)(s') = \eta\, Z(s',a,o) \int_{s \in S} T(s,a,s') b(s) ds \quad (1)$$

where $\eta$ is a normalization constant.

### 2.2 Linearization in Control and Estimation

It is well known that linearization (and Gaussian assumption for stochastic system models) only performs well when the system's non-linearity is "weak" [15]. The question is of course what does "weak" non-linearity means. Many measures have been proposed to answer this question. Most of these measures are designed for deterministic systems [3; 4; 10]. Some of them are based on the relative curvature [3], which is derived from the ratio between the second and first derivatives of the function, while others, e.g., [4], defines non-linearity measure based on the distance between the non-linear function and its nearest linearization. Few works have expanded these measures to stochastic systems. For instance, the work in [15] extends the approach of [4] to measure non-linearity based on the average distance between the non-linear function and the set of all possible linearization of the function. More recently, [9] proposes a different class of measures, which is based on the distance between distribution over states and its Gaussian approximation, rather than based on the non-linear function itself. A short survey on measures of non-linearity is available at [15].

Despite the many proposed measure of non-linearity, they are not well suited for motion planning under uncertainty, due to the difficulties in accounting the effect of obstacles to linearization. These non-linearity measures are based on the robot's dynamics model when operating in open space. Of course, obstacles can be incorporated to the robot's state space and dynamics models as constraints. However, this is difficult to do for robots with many dimensions. In fact the rise of probabilistic motion planning is exactly to avoid explicit construction of collision constraints / collision regions in the state space. Therefore, the actual effects of obstacle avoidance to the effectiveness of linearization is still relatively unknown. This paper presents our exploratory studies that attempt to understand the effects that obstacles have on the effectiveness of linearization in stochastic motion planning.

## 3 Methods for Comparison

Our comparison studies will use LQG-MP [5], an LQG based POMDP solver, and ABT [13], a general POMDP solver. ABT provides a base-line for the performance of methods that do not use linearization and Gaussian assumption as the environment becomes more cluttered.

### 3.1 LQG-MP

LQG-MP [5] formulates the motion planning problem as a linear-quadratic-Gaussian control problem [2]. Given a nominal trajectory $(s_t^*, a_t^*)_{t=0}^T$, where $s^* \in S$ and $a^* \in A$ the robot has to follow, the control problem is formulated as a cost-minimisation problem, such that deviations of the robot from the given trajectory are quatratically penalized. More specifically, the goal of the LQG-problem is to control a feedback controller that minimizes the cost function

$$E\left(\sum_{t=0}^{T} (s_t - s_t^*)^T C (s_t^* - s_t) + (a_t - a_t^*)^T D (a_t - a_t^*)\right) \quad (2)$$

where $C$ and $D$ are positive-definite weight matrices. It is well known that the LQR-feedback-controller that minimizes 2 for a linear system is the solution of a set of Ricatti difference equations. [5] shows that for a given nominal trajectory, the distributions of the robot's state and control inputs can be

estimated a-priori by linearizing the system around the trajectory and assuming a LQR control policy. These distributions can then be used to evaluate a given trajectory according to a planning objective (e.q. maximizing the likelihood of a robot to reach a given goal area inside an environment).

Given an LQR-feedback-controller that minimizes 2, [5] shows how to approximate *a-priori* the distribution of the system at each state of the nominal trajectory as a Gaussian distribution, assuming the system can be well linearized around the trajectory. Using these approximated distributions, the quality of a trajectory can be quickly evaluated according to a planning objective (e.q. maximizing the likelihood of a robot to reach a given goal area inside an environment). The idea of LQG-MP is to sample a large set of candidate trajectories for a given motion planning problem. At run-time, after selecting a trajectory according to the planning objective, LQG-MP uses LQR-feedback-controller as the control policy and a Kalman filter to update the belief estimate of the robot.

### 3.2 Adaptive Belief Tree

ABT [13] is a online POMDP solver based on Monte-Carlo tree search. From a given belief $b_t$, ABT aims to find an action $a^*$ that maximises the expected discounted future reward. In order to do so, ABT iteratively generates a belief tree. Each node in the tree corresponds to a belief, while the edges between nodes correspond to action-observation pairs. From the current belief $b_t$, which is maintained by a set of particles, ABT maintains a set of samples episodes. An episode consists of a sequence of state, action, observation and immediate reward quadruples $(s,a,o,r)$. To sample an episode, ABT samples a state $s_0$ from $b_t$, selects and action and uses a generative black-box model to generate an observation and immediate reward and the next state $s_1$. One advantage of this approach is that the exact belief dynamics don't have to be explicilty known.

These episodes are maintained within the belief tree. Since the belief nodes are represented by a set of particles, each node consists of the states of the corresponding quadruples of the corresponding episodes. In order to select an action from $b_t$, ABT approximates $\hat{Q}(b_t,a)$, which is the value of performing $a$ from $b_t$ and continuing optimally afterwards, using forward simulation and value backup. At the leaf-nodes, $Q(b,a)$ is approximated by a heuristic function. After the planning time is over, ABT selects and executes the action with the highest $Q$ and updates $b_t$ after receiving an observation.

## 4 Comparison Studies

In order to understand how cluttered environments affect the performance of motion planning algorithms based on linearization and Gaussian simplification, we compare LQG-MP and ABT using two problem scenarios: A car-like non-holonomic robot with $2^{nd}$ order dynamics driving on a flat $xy$-plane and a 4-degree-of-freedom holonomic manipulator operating inside a 3D-environment. In each problem scenario, the robot starts from a known initial state $s_0$ and has to reach a terminal state $s_T \in G$ where $G \subset S$ is the set of terminal states. Instead of looking at a fixed environment for the two problem scenarios, we investigate how ABT and LQG-MP perform in increasingly cluttered random environments when the motion- and observation uncertainty increases.

### 4.1 Car-like robot

In this scenario a car-like robot starts from a known initial state $s_0$ (lower-left in figure 1) and has to reach a goal-area in the upper-right corner of the map (green sphere in figure 1) while avoiding collision with the obstacles (red boxes).

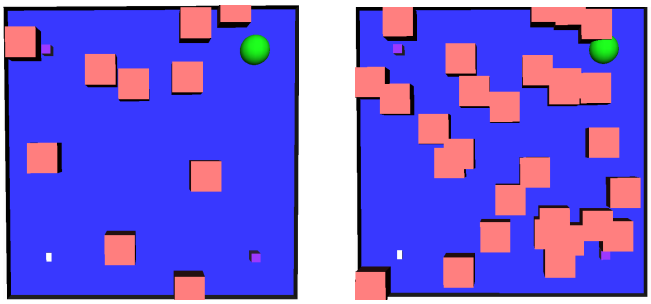

Figure 1: Two randomly sampled environment for the car-like robot with 10 obstacles and 30 obstacles (right)

The state of the robot is defined as a 4D vector $s_t = (x_t, y_t, \theta_t, v_t)^T$ where $x_t$ and $y_t$ is the position of the center of the robot on the $xy$-plane, $\theta_t$ the orientation and $v_t$ the linear velocity of the robot. The control input $a_t = (\alpha_t, \phi_t)^T$ is a 2D vector consisting of the acceleration $\alpha$ and the steering wheel angle $\phi$. We assume that the control input is subject to control noise $v_t = (\tilde{\alpha}_t, \tilde{\phi}_t)^T \sim N(0, \Sigma_v)$

The nonlinear stochastic dynamics of the robot is modelled as

$$s_{t+1} = \begin{bmatrix} x_t + \Delta t\, v cos\theta_t \\ y_t + \Delta t\, v sin\theta_t \\ \theta + \Delta t\ tan(\phi_t + \tilde{\phi})/d \\ v + \Delta t(\alpha_t + \tilde{\alpha}) \end{bmatrix}$$

where $\Delta t$ is the duration of a time step and $d$ the distance between the front and rear axles of the wheels.

The robot localizes itself with the help of two beacons located inside the environment. Suppose the beacons are located at $(\hat{x}_1, \hat{y}_1)$ and $(\hat{x}_2, \hat{x}_1)$. Then, the signals the robot receives from these two beacons is a function of the distance to them. Furthermore the robot receives information regarding its current velocity from a velocity sensor mounted on the robot. We assume that both sensor readings are disturbed by sensor noise $w_t$. More formally, the robot's observation model is defined as:

$$z_t = \begin{bmatrix} 1/((x_t - \hat{x}_1)^2 + (y_t - \hat{y}_1)^2 + 1) \\ 1/((x_t - \hat{x}_2)^2 + (y_t - \hat{y}_2)^2 + 1) \\ v_t \end{bmatrix} + w_t$$

where $w_t$ is drawn from a zero-mean Gaussian distribution, such that $w_t \sim N(0, \Sigma_w)$.

### 4.2 4DOF-manipulator

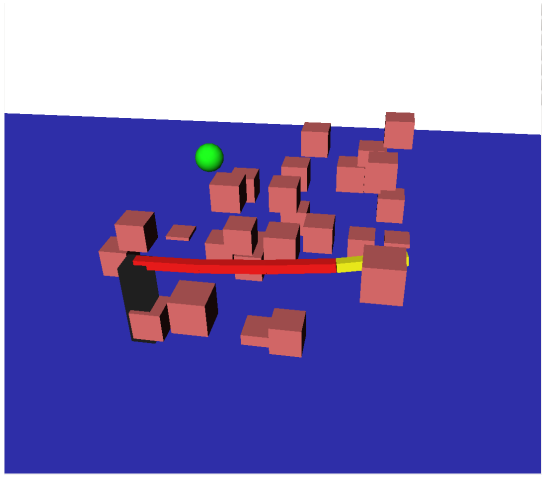

Figure 2: A randomly sampled manipulator environment with 30 obstacles. The manipulator consists of a fixed base (black cuboid) and 4 links connected by rotary joints. The yellow link visualizes the end-effector link. The green sphere is the goal area for the end-effector

In this set of experiments we consider a 4-degrees-of-freedom holonomic, articulated manipulator with a fixed base (black cuboid in figure 2). Starting from an initial state $x_0$, the goal of the planner is to move the manipulator to a state in which the end-effector (yellow link in figure 2) is inside a goal region inside the robot's workspace, avoiding self-collision and collisions with the obstacles in the environment. The state of the manipulator is defined as $s = (\theta, \dot{\theta})^T \in \mathbb{R}^8$, where $\theta$ is the vector of joint angles and $\dot{\theta}$ is the vector of rotational joint velocities. Using Lagrangian dynamics, the manipulator is modelled as a continuous-time stochastic system of the form:

$$\frac{\partial}{\partial t}\begin{bmatrix}\theta\\ \dot{\theta}\end{bmatrix} = \begin{bmatrix}\dot{\theta}\\ M(\theta)^{-1}(a+v-C(\theta,\dot{\theta})\dot{\theta}-N(\theta))\end{bmatrix} \quad (3)$$

where $M \in \mathbb{R}^{4\times4}$ is the inertia matrix, $C \in \mathbb{R}^{4\times4}$ the centrifugal and coriolis matrix and $N \in \mathbb{R}^4$ the vector of external forces acting on the joints. Here we assume that the external forces are gravity and viscous joint friction. The control input $a \in A \subset \mathbb{R}^4$ is the joint torques and is assumed to be disturbed by a 4-dimensional random error vector $v \sim N(0, \Sigma_v)$. In our experiments, we discretize the time to transform the continuous-time dynamics into a discrete-time system. Note also that although the error is Gaussian distributed, due to the non-linearity of the dynamics, the resulting belief estimate will generally not be Gaussian distributed.

The robot is equipped with two types of sensors. The first sensor measures the position of the end-effector in the robot's workspace. The second sensor measures the joint velocities. Suppose $g : \mathbb{R}^4 \mapsto \mathbb{R}^3$ is a function that maps the state of the robot to an end-effector position in the workspace and $w_t \sim N(0, \Sigma_w)$ is the error vector, then the observation model is defined as $z_t = [g(s_t), \dot{\theta}_t]^T + w_t$.

### 4.3 Experimental setup

To see the effect of increasingly cluttered environments on the performance of LQG-MP and ABT, the experiments conducted for this paper for both robots consist of 4 scenarios: An environment with 0, 10, 20 and 30 fixed-sized, randomly distributed box-shaped obstacles. Furthermore, we test each scenario using 2 motion and observation error values $err_{motion}, err_{obs} = 2.5$ and $err_{motion}, err_{obs} = 5.0$. $err_{motion}, err_{obs}$ are hereby defined as follows: Recall that in our experimental setting for both robots, the motion and observation noise terms are drawn from zero-mean multivariate Normal distributions with covariance matrices $\Sigma_v$ and $\Sigma_w$. $err_{motion}, err_{obs}$ serve as scaling parameters for the variance terms of the covariance matrices (the diagonal entries $\sigma_1, ..., \sigma_n$). In the case of the observaton error error covariance matrix, these variance parameters are defined as $\sigma_i = \left[\frac{|O_i|}{100} err_{obs}\right]^2$ where $|O_i|$ is the value range of the $i$-th dimension of $O$. Similarly the variance entries of the motion-error covariance matrix are defined as $\sigma_i = \left[\frac{|A_i|}{100} err_{motion}\right]^2$. For each type of scenario (0, 10, 20 and 30 random obstacles) we run 50 simulation runs for the two error values. In each simulation run we sample a new random environment with the respective number of rando obstacles. We hereby assume that once a new environment has been sampled, is is fully known to the robot.

Since the idea behind LQG-MP is to evaluate a large set of nominal trajectories for a given scenario, we sample a set of 500 trajectories for each random scenario which serve as candidate trajectories for LQG-MP. Throughout all the experiments, we use RRT [14] to sample the candidate trajectories. Here the planning objective is to maximise the the expected discounted reward when following a specific trajectory. We evaluate the 500 candidate trajectories using a simple Monte-Carlo approach: LQG-MP approximates the beliefs around a trajectory with Gaussian distributions. By sampling these distributions using 500 samples per belief, we can approximate the expected discounted reward of the trajectory. The reward function used throughout the experiments is defined as $R(s_t) = 1000$ if $s_t$ is a terminal state, $R(s_t) = -500$ when $s_t$ collides with an obstacle, or, in case-of the manipulator robot, collides with itself, and $R(s_t) = -1$ else. The same reward function is used for ABT throughout the experiments. Since ABT is an online planner we give 1 second of planning time for each planning step.

#### Results and Discussion

Figure 3 presents the mean rewards achieved by both algorithms in the car-like scenario as the number of obstacles increase. It can be seen that, as the number of obstacles increase

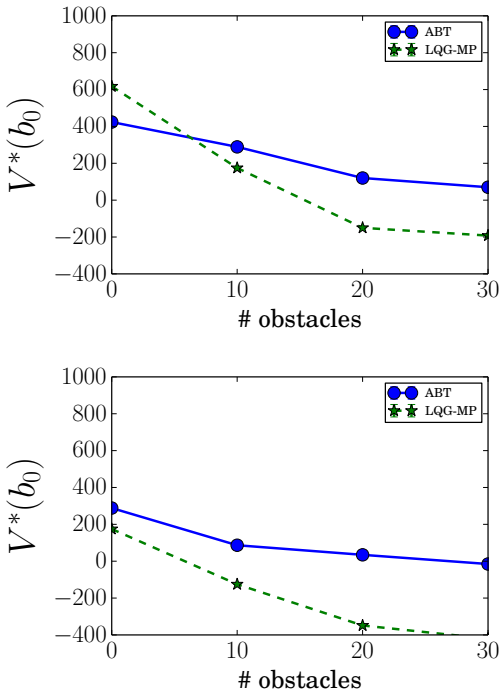


Figure 3: This plot show the mean reward of 50 simulation runs achieved for the car-like robot by ABT and LQG-MP with $err_{obs} = err_{motion} = 2.5$ (top figure) and $err_{obs} = err_{motion} = 5.0$ (bottom figure). The $x$-axis shows the number of randomly sampled obstacles for each simulation run. The $y$-axis shows the mean reward achieved over 50 simulation runs starting from $b_0$

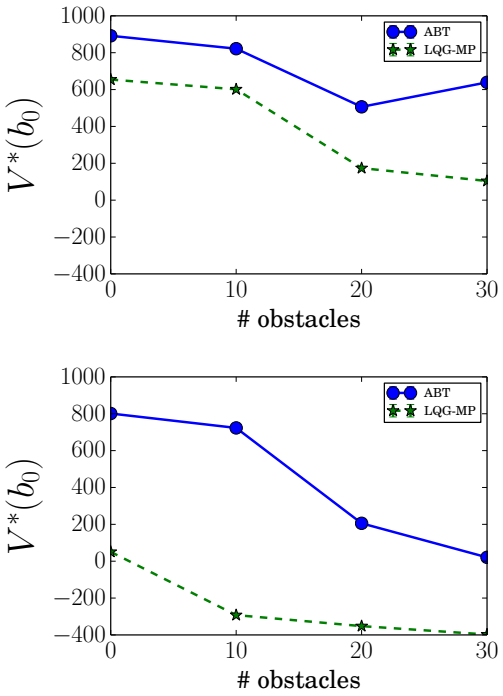


Figure 4: This plot show the mean reward of 50 simulation runs achieved for the manipulator robot by ABT and LQG-MP with $err_{obs} = err_{motion} = 2.5$ (top figure) and $err_{obs} = err_{motion} = 5.0$ (bottom figure). The $x$-axis shows the number of randomly sampled obstacles for each simulation run. The $y$-axis shows the mean reward achieved over 50 simulation runs starting from $b_0$

and the environment get more cluttered, LQG-MP seems to be more sensitive to increasingly cluttered environments. Intuitively, as the environment becomes more cluttered and the robot operates in the proximity of obstacles, the Gaussian assumption can be a particular poor approximation of the robot's belief. In what follows is that the LQR-control policy is suboptimal with respect to the true belief of the robot. Furthermore, the LQR control policy is is a minimizer for the quadratic cost function 2 which does not include cost terms that are depended on the environment. Hence, during runtime, the LQR controller objective is to follow the nominal trajectory as closely as possible, even though this possibly yields dangerous situations near obstacles. Looking at the results of the manipulator scenario in figure 4, it becomes clear that the advantage of ABT over LQG-MP increases, as the uncertainty increases. This is not surprising, since the manipulator as modelled in the experiments is a highly non-linear system for which linearization only works for small errors. A possible interresting future venue could be to quantify how well a stochastic system is linerizable and, following that, when a linearization-based solver is likely to compute good motion strategies.

## 5 Conclusion

This paper presents our comparative study in understanding the effect of linearization and Gaussian simplifications in motion planning under uncertainty. In particular, our study indicates that the effectiveness of linearization and Gaussian simplifications decreases as the environment becomes more cluttered. Future work abounds. First, more exhaustive comparison studies are needed to better understand the issue. Theoretically, it is useful to design a non-linearity measure suited for motion planning under uncertainty. Measures that can be computed efficiently could also improve existing capabilities in motion planning under uncertainty. Furthermore, such a measure may help us in deciding what type of linearization methods to use for particular types of problems. Some part of these possible extensions (i.e., a more suitable non-linearity measure) will be available in [reference omitted to comply with the double-blind requirement].

## References

[1] A.-A. Agha-Mohammadi, S. Chakravorty, and N. M. Amato. Firm: Sampling-based feedback motion planning under motion uncertainty and imperfect measure-

ments. *The International Journal of Robotics Research*, 2013.

[2] M. Athans. The role and use of the stochastic linear-quadratic-gaussian problem in control system design. *IEEE transactions on automatic control*, 16(6):529–552, 1971.

[3] D. M. Bates and D. G. Watts. Relative curvature measures of nonlinearity. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–25, 1980.

[4] E. Beale. Confidence regions in non-linear estimation. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 41–88, 1960.

[5] J. Berg, P. Abbeel, and K. Goldberg. LQG-MP: Optimized Path Planning for Robots with Motion Uncertainty and Imperfect State Information. In *RSS*, 2010.

[6] J. Berg, D. Wilkie, S. Guy, M. Niethammer, and D. Manocha. LQG-Obstacles: Feedback Control with Collision Avoidance for Mobile Robots with Motion and Sensing Uncertainty. In *ICRA*, 2012.

[7] D. P. Bertsekas, D. P. Bertsekas, D. P. Bertsekas, and D. P. Bertsekas. *Dynamic programming and optimal control*, volume 1. Athena Scientific Belmont, MA, 1995.

[8] A. W. Drake. *Observation of a Markov process through a noisy channel*. PhD thesis, Massachusetts Institute of Technology, 1962.

[9] J. Duník, O. Straka, and M. Šimandl. Nonlinearity and non-gaussianity measures for stochastic dynamic systems. In *Information Fusion (FUSION), 2013 16th International Conference on*, pages 204–211. IEEE, 2013.

[10] K. Emancipator and M. H. Kroll. A quantitative measure of nonlinearity. *Clinical chemistry*, 39(5):766–772, 1993.

[11] M. Horowitz and J. Burdick. Interactive Non-Prehensile Manipulation for Grasping Via POMDPs. In *ICRA*, 2013.

[12] L. Kaelbling, M. Littman, and A. Cassandra. Planning and acting in partially observable stochastic domains. *AI*, 101:99–134, 1998.

[13] H. Kurniawati and V. Yadav. An online POMDP solver for uncertainty planning in dynamic environment. In *ISRR*, 2013.

[14] S. M. Lavalle and J. J. Kuffner Jr. Rapidly-exploring random trees: Progress and prospects. In *Algorithmic and Computational Robotics: New Directions*. Citeseer, 2000.

[15] X. R. Li. Measure of nonlinearity for stochastic systems. In *Information Fusion (FUSION), 2012 15th International Conference on*, pages 1073–1080. IEEE, 2012.

[16] G. E. Monahan. State of the arta survey of partially observable markov decision processes: theory, models, and algorithms. *Management Science*, 28(1):1–16, 1982.

[17] S. Prentice and N. Roy. The belief roadmap: Efficient planning in linear pomdps by factoring the covariance. In *Robotics Research*, pages 293–305. Springer, 2010.

[18] K. Seiler, H. Kurniawati, and S. Singh. An online and approximate solver for pomdps with continuous action space. In *ICRA*, 2015.

[19] D. Silver and J. Veness. Monte-Carlo Planning in Large POMDPs. In *NIPS*, 2010.

[20] A. Somani, N. Ye, D. Hsu, and W. S. Lee. DESPOT: Online POMDP planning with regularization. In *NIPS*, pages 1772–1780, 2013.

[21] W. Sun, S. Patil, and R. Alterovitz. High-frequency replanning under uncertainty using parallel sampling-based motion planning. *IEEE Transactions on Robotics*, 31(1):104–116, 2015.

[22] S. Temizer, M. Kochenderfer, L. Kaelbling, T. Lozano-Pérez, and J. Kuchar. Unmanned Aircraft Collision Avoidance Using Partially Observable Markov Decision Processes. Project Report ATC-356, MIT Lincoln Laboratory, Advanced Concepts Program, Lexington, Massachusetts, USA, September 2009.