LQG-Obstacles: Feedback Control with Collision Avoidance for Mobile Robots with Motion and Sensing Uncertainty

Jur van den Berg David Wilkie Stephen J. Guy Marc Niethammer Dinesh Manocha

***Abstract*— This paper presents *LQG-Obstacles*, a new concept that combines linear-quadratic feedback control of mobile robots with guaranteed avoidance of collisions with obstacles. Our approach generalizes the concept of Velocity Obstacles [3] to any robotic system with a linear Gaussian dynamics model. We integrate a Kalman filter for state estimation and an LQR feedback controller into a closed-loop dynamics model of which a higher-level control objective is the "control input". We then define the LQG-Obstacle as the set of control objectives that result in a collision with high probability. Selecting a control objective outside the LQG-Obstacle then produces collision-free motion. We demonstrate the potential of LQG-Obstacles by safely and smoothly navigating a simulated quadrotor helicopter with complex non-linear dynamics and motion and sensing uncertainty through three-dimensional environments with obstacles and narrow passages.**

I. Introduction

Planning under uncertainty has received considerable attention in robotics in recent years. Motion uncertainty (due to external disturbances) and imperfect state information (due to partial and noisy measurements) arise in most real-world robotic tasks. This is especially true for highly dynamic mobile robots that must reach a target configuration while avoiding collisions with obstacles, such as quadrotor helicopters [11]. Traditional planning approaches [15], [9] may not suffice in these cases, as they assume deterministic motion and full knowledge of the state, and often produce jerky paths due to the random nature of the algorithm.

Feedback controllers, on the other hand, can compensate for motion and sensing uncertainty while smoothly controlling a mobile robot towards a target configuration, either directly or by tracking a pre-planned path. The linear-quadratic Gaussian (LQG) controller does so optimally for linear systems with Gaussian noise [2], and is widely used for non-linear models as well [27]. LQG control does not, however, account for the risk of colliding with obstacles in the environment.

To address this shortcoming, we present the novel concept of *LQG-Obstacles* for combining LQG feedback control with collision avoidance. LQG-Obstacles generalize Velocity Obstacles [3] to any linear Gaussian system. Our approach integrates a Kalman filter for state estimation and an LQR feedback controller into a closed-loop dynamics model of which a higher-level control objective (a target configuration) is the "control input". LQG-Obstacles then identify all target configurations to which a mobile robot can safely be controlled using the LQG controller. For deterministic dynamics models, the LQG-Obstacle *guarantees* that the robot will not collide with an obstacle. We consider this to be a special case and refer to the LQG-Obstacle as an *LQR-Obstacle* for this class of models. For stochastic models, the general case, the LQG-Obstacle bounds the a priori probability of colliding with obstacles to a user-specified limit, explicitly accounting for motion and sensing uncertainty.

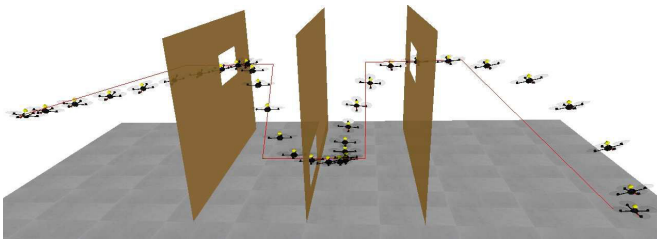


Fig. 1. The *3-Window* scenario for our quadrotor simulation experiments. The red line is the guiding path. Our approach smoothly controls the quadrotor (shown at 4Hz) through the windows without colliding with the walls. Videos are available at `http://gamma.cs.unc.edu/CA/LQGObs/`.

Our concept can be used to safely and smoothly navigate a robot with complex dynamics through an environment with obstacles, by continually selecting a target configuration allowed by the computed LQG-Obstacle. The LQG controller then determines the corresponding control input. While our approach is designed for robots with linear dynamics and observation models, it can also be applied to robots with non-linear models, by continually linearizing them around the nearest steady state. The LQG-Obstacle automatically adapts to the local amount of motion and sensing uncertainty: it forces the robot to choose safer control objectives if this uncertainty is high, and allows for more aggressive motion when there is less uncertainty. In contrast to existing collision avoidance and (re)planning approaches, our approach (i) produces smooth motion, (ii) works at real-time rates even for robots with high-dimensional state spaces, (iii) is applicable to robots requiring high-frequency control feedback loops, and (iv) explicitly considers motion and sensing uncertainty.

While our approach naturally extends to moving obstacles, we focus on environments with static obstacles in this paper. We assume that the geometry of the obstacles in the robot's configuration space are given explicitly. Further, we assume the dynamics and observation models of the robot to be given, and that their uncertainty can be modeled by known Gaussian distributions. We implemented our approach for

Jur van den Berg is with the School of Computing, University of Utah. E-mail: `berg@cs.utah.edu`.

David Wilkie, Stephen J. Guy, Marc Niethammer, and Dinesh Manocha are with the Department of Computer Science, University of North Carolina at Chapel Hill. E-mail: {`wilkie, sjguy, mn, dm`}`@cs.unc.edu`.

This work was supported by ARO Contract W911NF-10-1-0506, NSF awards 1000579 and 1117127.

two illustrative cases. The first is a linear mobile robot in the plane whose acceleration can be controlled. The second is a quadrotor helicopter with a thirteen-dimensional state space and non-linear dynamics flying in three-dimensional workspaces. We validate our approach using simulation experiments with synthetic motion and sensor noise in environments with obstacles and narrow corridors. Our results suggest that LQG-Obstacles can work well for safe, real-time LQG control of mobile robots towards a goal configuration amidst obstacles (see Fig. 1).

The remainder of this paper is organized as follows. In Section II we discuss related work. In Section III we introduce LQR-Obstacles for robots with deterministic dynamics. In Section IV we extend this to LQG-Obstacles for robots with stochastic dynamics and observation models. We report simulation results in Section V and conclude in Section VI.

II. Background and Related Work

A. Planning and Control under Uncertainty

The problem of planning under motion and sensing uncertainty is most generally described as a partially-observable Markov decision process (POMDP) [10]. POMDPs provide a formulation of the objective to optimize the overall probability of arriving at a goal configuration without collisions. Unfortunately, in their general form, POMDPs are known to be of extreme complexity, which makes it challenging to solve them for large or high-dimensional state spaces [18]. Most approaches that approximate an optimal solution to POMDPs rely on discretization or random sampling of the belief space, or a combination of both [12], [20], [26]. However, discretization error and high computational costs may prohibit their application to mobile robots with high-frequency control feedback loops.

Another class of algorithms assume linear(ized) Gaussian dynamics and observation models. LQG-MP [28] calculates the probability of successful execution of a given path based on computed a-priori probability distributions. Belief space planning approaches based on the LQG model [8], [21], [22] attempt to optimize the likelihood of arriving at the goal, but do not take into account the probability of colliding with obstacles. More recent work [30] does account for obstacles, but its applicability is limited to robots with low-dimensional state spaces. These approaches focus on the ability to generate information gathering actions as to minimize uncertainty about the robot's state. In contrast, our approach does not look ahead to select actions that provide most information, rather it bounds the probability of collision with obstacles based on the local sensing and acting capabilities of the robot. This goal is complementary to that of belief space planning approaches.

Our work shares similarities with the feedback motion planning approach of LQR-trees [25], which cover the state space with stabilizing LQR controllers around paths in a tree that guarantees the robot to reach the goal from any point in the state space. Our approach may complement LQR-trees, as they do not explicitly account for the presence of obstacles in combination with motion and sensing uncertainty. Our work is also related to model-predictive control approaches [23], which do account for constraints on the state in an optimal control formulation. However, they often require solving high-dimensional mixed-integer programs in every control cycle, even if the constraints are linear and convex.

B. Collision Avoidance

Our approach generalizes concepts in reactive collision avoidance, in particular *Velocity Obstacles* [3]. Velocity obstacles identify all velocities that lead to a collision at any time in the future. Velocities can then be chosen outside of the velocity obstacle to ensure that no collision will occur. Stochastic variants of the velocity obstacle deal with motion uncertainty by enlarging the velocity obstacle [24] or consider uncertainty in the motion of the obstacles [6]. A limitation of approaches based on velocity obstacles is that they only work for robots whose velocity can directly be controlled. Extensions exist for robots of which the acceleration can be controlled [29] and for car-like robots [17], [32]. Our approach generalizes these methods to any linear Gaussian system.

An alternative to feedback control and reactive collision avoidance is continual (partial) replanning [19], potentially in combination with dynamic windows [4] or ICS checking [5]. The latter concept has recently been extended to account for motion uncertainty of the obstacles [1]. The achievable planning frequency and path quality may not be high enough for highly dynamic systems such as quadrotors, though.

C. Notation

We use the following notational conventions in this paper. Vector sets $\mathcal{A}$ are denoted using calligraphics, vectors $\mathbf{a}$ are denoted using boldface, matrices A are denoted using upper case italics, and scalars a are denoted in lower-case italics.

Scalar and matrix multiplication of sets are defined as:

$$a\mathcal{X} = \{a\mathbf{x} \mid \mathbf{x} \in \mathcal{X}\}, \qquad A\mathcal{X} = \{A\mathbf{x} \mid \mathbf{x} \in \mathcal{X}\}. \tag{1}$$

The Minkowski sum of two sets is defined as:

$$\mathcal{X} \oplus \mathcal{Y} = \{\mathbf{x} + \mathbf{y} \mid \mathbf{x} \in \mathcal{X}, \mathbf{y} \in \mathcal{Y}\}. \tag{2}$$

It follows that $\mathcal{X} \oplus \{\mathbf{x}\}$ denotes a translation of a set $\mathcal{X}$ by a vector $\mathbf{x}$.

III. LQR-Obstacles for Deterministic Systems

In this section we discuss how LQR feedback control and collision avoidance are combined for robots with deterministic linear dynamics and perfect state information. We first discuss LQR control and derive the closed-loop linear dynamics. We then define the *LQR-Obstacle* for collision-free LQR control.

A. LQR Feedback Control

Let $\mathcal{X} \subset \mathbb{R}^x$ be the *state space* of the robot, which is the space of vectors containing all information relevant for the motion of the robot, and let $\mathcal{C} \subset \mathbb{R}^c$ be the *configuration space* of the robot ($c \leq x$), which is the space of vectors containing all information relevant for the

geometric appearance of the robot in the workspace. Let a given matrix $C \in \mathbb{R}^{c \times x}$ map the state $\mathbf{x} \in \mathcal{X}$ of the robot to its corresponding configuration $C\mathbf{x}$ in $\mathcal{C}$. Let $\mathcal{U} \subset \mathbb{R}^u$ be the control input space of the robot.

Let the dynamics of the robot be given by the deterministic linear model, which we assume is formally *controllable*:

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u}, \tag{3}$$

where $\mathbf{x} \in \mathcal{X}$ and $\mathbf{u} \in \mathcal{U}$ are the state vector and control input vector, respectively, of the robot, and $A \in \mathbb{R}^{x \times x}$ and $B \in \mathbb{R}^{x \times u}$ are given constant matrices.

Let $\mathbf{c} \in \mathcal{C}$ denote a target configuration the robot wishes to reach. For systems with linear dynamics, an *LQR feedback controller* can optimally control the robot towards this target state if a quadratic cost function is specified that trades-off reaching the target quickly, versus not applying extreme control inputs [2]:

$$\int_0^\infty \left((C\mathbf{x} - \mathbf{c})^T Q (C\mathbf{x} - \mathbf{c}) + \mathbf{u}^T R \mathbf{u}\right) \mathrm{d}t, \tag{4}$$

where $Q \in \mathbb{R}^{c \times c}$ and $R \in \mathbb{R}^{u \times u}$ are given constant weight matrices, for which $Q = Q^T \geq 0$ and $R = R^T > 0$.

The feedback control policy that minimizes this cost function is given by:

$$\mathbf{u} = -L\mathbf{x} + E\mathbf{c}, \tag{5}$$

where

$$L = R^{-1}B^T S, \quad E = R^{-1}B^T (BL - A)^{-T} C^T Q, \tag{6}$$

with S being the positive-definite solution to the continuous-time algebraic Riccati equation:

$$A^T S + SA - SBR^{-1}B^T S + C^T Q C = 0. \tag{7}$$

This is the standard continuous-time infinite-horizon LQR derivation [2]. Note that L and E are constant and can be computed given the matrices A, B, Q, and R [16].

We create the *closed-loop* dynamics of the robot by substituting Eq. (5) into Eq. (3), which gives:

$$\dot{\mathbf{x}} = \tilde{A}\mathbf{x} + \tilde{B}\mathbf{c}, \tag{8}$$

with

$$\tilde{A} = A - BL, \qquad \tilde{B} = BE. \tag{9}$$

The target configuration $\mathbf{c}$ is the higher-level "control input" of the closed-loop linear dynamics. We use the closed-loop dynamics to define LQR-Obstacles below.

B. Constructing LQR-Obstacles

Let $\mathcal{O} \subset \mathcal{C}$ denote the forbidden region in the configuration space of the robot that is occupied by obstacles. Then, the *LQR-Obstacle* for the robot is defined as:

Definition 1 Given the current state $\mathbf{x}$, the *LQR-Obstacle* $\mathcal{LQR}(\mathbf{x}) \subset \mathcal{C}$ is the set of target configurations $\mathbf{c}$ that let the robot collide with an obstacle at some point in time when the LQR control policy is used to control the robot to $\mathbf{c}$.

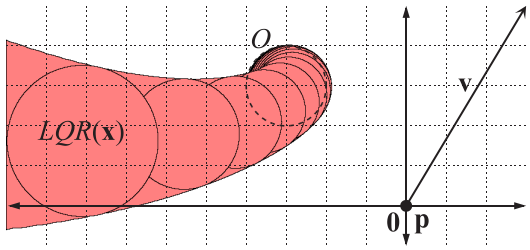


Fig. 2. The LQR-Obstacle $\mathcal{LQR}(\mathbf{x})$ shown as a union of transformed configuration-space obstacles for a planar robot of which the acceleration can be controlled (see Section V-B) with current state $\mathbf{x} = (\mathbf{p}, \mathbf{v})$. The configuration-space obstacle $\mathcal{O}$ is a disc shown by the dashed circle. The LQR-Obstacle consists of all target configurations $\mathbf{c}$ that result in a collision when an LQR controller is used to control the robot to that configuration.

We construct the LQR-Obstacle as follows. Integrating the closed-loop dynamics of Eq. (8) assuming a constant target configuration $\mathbf{c}$ gives:

$$\mathbf{x}(t) = F(t)\mathbf{x}(0) + G(t)\mathbf{c}, \tag{10}$$

with

$$F(t) = \exp(t\tilde{A}), \quad G(t) = \tilde{A}^{-1}(\exp(t\tilde{A}) - I)\tilde{B}. \tag{11}$$

Then, the robot will collide with an obstacle at time t if:

$$\begin{aligned} & C\mathbf{x}(t) \in \mathcal{O}, \\ \iff & CF(t)\mathbf{x}(0) + CG(t)\mathbf{c} \in \mathcal{O} \\ \iff & \mathbf{c} \in (CG(t))^{-1}(\mathcal{O} \oplus \{-CF(t)\mathbf{x}(0)\}), \end{aligned} \tag{12}$$

where we assume that $CG(t) \in \mathbb{R}^{c \times c}$ is invertible. Hence, the LQR-Obstacle $\mathcal{LQR}(\mathbf{x})$ is defined as a union of transformed configuration-space obstacles (see Fig. 2):

$$\mathcal{LQR}(\mathbf{x}) = \bigcup_{t>0} (CG(t))^{-1}(\mathcal{O} \oplus \{-CF(t)\mathbf{x}\}). \tag{13}$$

The definition of the LQR-Obstacle implies that if the robot chooses its target configuration $\mathbf{c}$ outside $\mathcal{LQR}(\mathbf{x})$, the robot will not collide with any of the obstacles while it is controlled towards $\mathbf{c}$. The above formulation generalizes earlier collision avoidance concepts, such as velocity obstacles [3] and acceleration obstacles [29], to systems with arbitrary linear dynamics.

C. Properties and Complexity of LQR-Obstacles

If the configuration space obstacle $\mathcal{O}$ is of $O(1)$ geometric complexity, then $\mathcal{LQR}(\mathbf{x})$ is of $O(1)$ complexity as well. A closed-form expression can be derived for the boundary of the LQR-Obstacle if $\mathcal{O}$ is circular or a line-segment, and $CG(t)$ is a scalar matrix [29].

Further, if $\mathcal{O} = \mathcal{O}_1 \cup \mathcal{O}_2$, then $\mathcal{LQR}(\mathbf{x}) = \mathcal{LQR}_{\mathcal{O}_1}(\mathbf{x}) \cup \mathcal{LQR}_{\mathcal{O}_2}(\mathbf{x})$. It follows that if $\mathcal{O}$ consists of $O(n)$ geometric primitives of $O(1)$ complexity each, the LQR-Obstacle for $\mathcal{O}$ is a union of $O(n)$ primitive LQR-Obstacles. This union has a worst-case complexity of $O(n^2)$, but we do not suspect that this bound is tight: the LQR-Obstacles may observe a *pseudo-disc* property which would allow for a lower total complexity. We leave this as an open question.

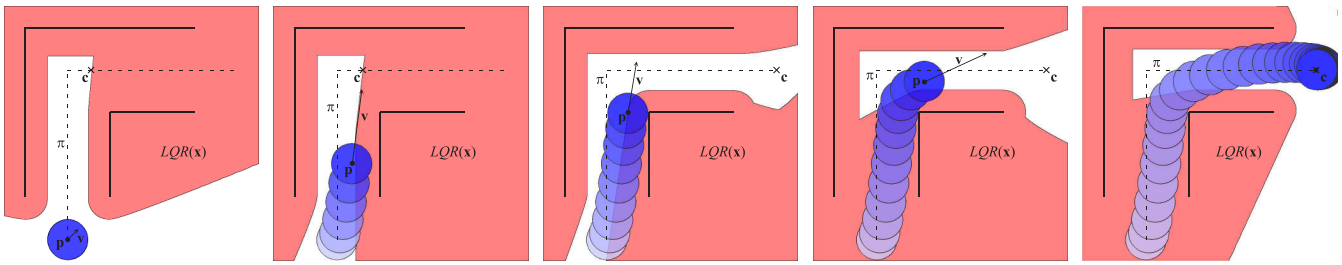


Fig. 3. Navigating a circular robot whose acceleration can be controlled (see Section V-B) through a narrow corridor using LQR-Obstacles. The workspace obstacles are shown using thick lines. The guiding path π is shown by a dashed line. In each frame, the LQR-Obstacle $\mathcal{LQR}(\mathbf{x})$ for the particular state $\mathbf{x} = (\mathbf{p}, \mathbf{v})$ is shown. The valid configuration $\mathbf{c}$ farthest along the guiding path is chosen as target configuration.

D. Collision-Free Feedback Control with LQR-Obstacles

LQR-Obstacles can be used to safely control a robot among obstacles in a continual cycle with a small time step Δt: in each time step, the LQR-Obstacle $\mathcal{LQR}(\mathbf{x})$ is computed, and a target configuration $\mathbf{c} \notin \mathcal{LQR}(\mathbf{x})$ is selected. The control input $\mathbf{u}$ that is applied is then given by Eq. (5). Note that the frequency of selecting a new target configuration $\mathbf{c}$ may well be lower than the LQR control frequency.

If a *goal* configuration $\mathbf{c}^\star$ is given, one may continually select the target configuration $\mathbf{c} \notin \mathcal{LQR}(\mathbf{x})$ that is closest to $\mathbf{c}^\star$. For circular configuration space obstacles $\mathcal{O}$ (e.g. Fig. 2), this will let the robot eventually reach the goal. For more generally shaped obstacles, however, this approach may easily lead the robot into a local minimum. A possible alternative in this case is to define a *guiding path* $\pi : [0, 1] \longrightarrow \mathcal{C} \setminus \mathcal{O}$ in the free configuration space, with $\pi(1) = \mathbf{c}^\star$, that indicates the global direction of motion of the robot. The robot may then continually select the target configuration farthest along π that is outside the LQR-Obstacle, i.e. $\mathbf{c} = \pi(\max\{s \,|\, \pi(s) \notin \mathcal{LQR}(\mathbf{x})\})$. Note that the guiding path π need not satisfy any differential constraints; a series of waypoints suffices. It can therefore easily be planned or constructed, e.g. by extracting it from a roadmap or tree covering the free configuration space [15], [9]. The LQR controller ensures that control inputs are chosen that result in smooth motion of the robot (see Fig. 3).

IV. LQG-Obstacles for Stochastic Systems

Above, we have assumed that the motion of the robot is deterministic, and that the robot has perfect information about its state. Here, we extend the method to deal with uncertainty in both the robot's dynamics and sensing. We will first discuss LQG control and derive the closed-loop linear Gaussian dynamics, and then define *LQG-Obstacles* for LQG control with bounded probability of collisions with obstacles. We discuss its application to non-linear systems as well.

A. LQG Control with State Estimation

Let the dynamics and observation models of the robot be given by the following linear Gaussian system:

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u} + \mathbf{m}, \qquad \mathbf{m} \sim \mathcal{N}(\mathbf{0}, M), \tag{14}$$

$$\mathbf{z} = H\mathbf{x} + \mathbf{n}, \qquad \mathbf{n} \sim \mathcal{N}(\mathbf{0}, N), \tag{15}$$

where Eq. (14) is similar to (3), except that the motion of the robot is corrupted by noise $\mathbf{m} \in \mathbb{R}^x$ drawn from an independent zero-mean Gaussian distribution with given constant variance $M \in \mathbb{R}^{x \times x}$. In the observation model, $\mathbf{z} \in \mathbb{R}^z$ is the vector of sensor measurements, and $H \in \mathbb{R}^{z \times x}$ is a given constant matrix. The sensor measurements are corrupted by noise $\mathbf{n} \in \mathbb{R}^z$ drawn from an independent zero-mean Gaussian distribution with given constant variance $N \in \mathbb{R}^{z \times z}$.

Let the control cost function be as in Eq. (4), given a target configuration $\mathbf{c} \in \mathcal{C}$ the robot wishes to reach. For linear Gaussian systems, an *LQG controller* is optimal. An LQG controller uses an LQR feedback controller in parallel with a Kalman filter for state estimation. The Kalman filter provides an optimal estimate $\hat{\mathbf{x}}$ of the state $\mathbf{x}$, which evolves given sensor measurements $\mathbf{z}$ as [7]:

$$\dot{\hat{\mathbf{x}}} = A\hat{\mathbf{x}} + B\mathbf{u} + K(\mathbf{z} - H\hat{\mathbf{x}}), \tag{16}$$

where K is the Kalman gain, which is given by:

$$K = PH^T N^{-1}, \tag{17}$$

where P is the variance of the state $\mathbf{x}$ given the state estimate $\hat{\mathbf{x}}$. Since our dynamics and observation model are stationary (i.e. the matrices A, H, M, and N are constant), this variance converges over time to the positive-definite solution of the continuous-time algebraic Riccati equation:

$$AP + PA^T + M - PH^T N^{-1} HP = 0. \tag{18}$$

Hence, the Kalman gain K is constant, and can be computed given the matrices A, H, M, and N [16].

Through the separation principle [2], the LQR control policy can be derived independently from the state estimator, and is therefore the same as in Eq. (5), with the difference that the state estimate $\hat{\mathbf{x}}$ is used instead of the (unknown) true state $\mathbf{x}$:

$$\mathbf{u} = -L\hat{\mathbf{x}} + E\mathbf{c}, \tag{19}$$

with L and E as defined in Eq. (5).

To create the closed-loop dynamics that incorporates both the state estimation and the feedback controller, we define an augmented state $\mathbf{y}$ that contains both $\mathbf{x}$ and $\hat{\mathbf{x}}$, for the true state $\mathbf{x}$ and the state estimate $\hat{\mathbf{x}}$ evolve as functions of each other [28]. Substituting Eq. (19) into (14) and Eqs. (15) and (19) into (16) gives:

$$\dot{\mathbf{y}} = \tilde{A}\mathbf{y} + \tilde{B}\mathbf{c} + \tilde{\mathbf{m}}, \qquad \tilde{\mathbf{m}} \sim \mathcal{N}(\mathbf{0}, \tilde{M}), \tag{20}$$

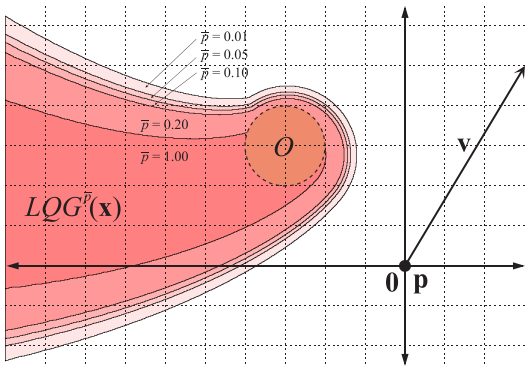


Fig. 4. The LQG-Obstacle $\mathcal{LQG}^{\bar{p}}(\hat{\mathbf{x}})$ in the same configuration as in Fig. 2 accounting for motion and sensing uncertainty of the robot (see Section V-B). The conservative approximation of Eq. (25) is shown for various values of the probability bound $\bar{p}$. For $\bar{p} = 1$, it is equivalent to the LQR-Obstacle.

with

$$\mathbf{y} = \begin{bmatrix} \mathbf{x} \\ \hat{\mathbf{x}} \end{bmatrix}, \qquad \tilde{A} = \begin{bmatrix} A & -BL \\ KH & A - BL - KH \end{bmatrix},$$
$$\tilde{B} = \begin{bmatrix} BE \\ BE \end{bmatrix}, \qquad \tilde{M} = \begin{bmatrix} M & 0 \\ 0 & KNK^T \end{bmatrix}. \tag{21}$$

Also in this case the target configuration $\mathbf{c}$ is the "control input" of the closed-loop linear Gaussian dynamics.

B. Constructing LQG-Obstacles

We now follow a similar approach as in Section III-B to define LQG-Obstacles.

Definition 2 Given the current state estimate $\hat{\mathbf{x}}$, the *LQG-Obstacle* $\mathcal{LQG}^{\bar{p}}(\hat{\mathbf{x}}) \subset \mathcal{C}$ for probability bound $\bar{p}$ is defined as the set of target configurations $\mathbf{c}$ for which there is a time $t > 0$ at which the probability that the robot collides with an obstacle is greater than $\bar{p}$ when LQG control is used to control the robot to $\mathbf{c}$.

We construct the LQG-Obstacle as follows. Integrating the closed-loop stochastic dynamics of Eq. (20) given a target configuration $\mathbf{c}$ gives $\mathbf{y}(t) \sim \mathcal{N}(\hat{\mathbf{y}}(t), Y(t))$, with:

$$\hat{\mathbf{y}}(t) = F(t)\hat{\mathbf{y}}(0) + G(t)\mathbf{c}, \tag{22}$$
$$Y(t) = F(t)Y(0)F^T(t) + \int_0^t F(\tau)\tilde{M}F^T(\tau)\,\mathrm{d}\tau, \tag{23}$$

where $F(t)$ and $G(t)$ are as in Eq. (10) for the matrices $\tilde{A}$ and $\tilde{B}$ of Eq. (21). Since the true state $\mathbf{x}$ is unknown, the initial conditions are $\hat{\mathbf{y}}(0) = \left[\begin{smallmatrix} \hat{\mathbf{x}} \\ \hat{\mathbf{x}} \end{smallmatrix}\right]$ and $Y(0) = \left[\begin{smallmatrix} P & 0 \\ 0 & 0 \end{smallmatrix}\right]$, where $\hat{\mathbf{x}}$ is the current state estimate. Recall that P is the variance of the true state $\mathbf{x}$ given its estimate $\hat{\mathbf{x}}$ (see Eq. (18)).

To map the augmented state $\mathbf{y}$ to the configuration of the robot, we define the augmented matrix $\tilde{C} = \begin{bmatrix} C & 0 \end{bmatrix}$. Now, following a similar derivation as in Eq. (12), the robot will collide with an obstacle at a specific time t if:

$$\tilde{C}\mathbf{y}(t) \in \mathcal{O} \iff \tilde{\mathbf{c}} \in (\tilde{C}G(t))^{-1}(\mathcal{O} \oplus \{-\tilde{C}F(t)\hat{\mathbf{y}}(0)\}), \tag{24}$$

where $\tilde{\mathbf{c}} \sim \mathcal{N}(\mathbf{c}, \Sigma(t))$, with $\Sigma(t) = (\tilde{C}G(t))^{-1}\tilde{C}Y(t)\tilde{C}^T \cdot (\tilde{C}G(t))^{-T}$. Let $\mathcal{E}^p(\Sigma)$ denote the contour ellipsoid of a zero-mean Gaussian distribution with variance Σ that contains a fraction $1 - p$ of its instances. Then, the probability that the robot will collide with an obstacle at time t is *less* than $\bar{p}$ if $\mathbf{c} \notin (\tilde{C}G(t))^{-1}(\mathcal{O} \oplus \{-\tilde{C}F(t)\hat{\mathbf{y}}(0)\}) \oplus \mathcal{E}^{\bar{p}}(\Sigma(t))$. Hence, a conservative approximation of the LQG-Obstacle can be constructed as (see Fig. 4):

$$\mathcal{LQG}^{\bar{p}}(\hat{\mathbf{x}}) \subset \bigcup_{t>0} (\tilde{C}G(t))^{-1}(\mathcal{O} \oplus \{-\tilde{C}F(t)\left[\begin{smallmatrix} \hat{\mathbf{x}} \\ \hat{\mathbf{x}} \end{smallmatrix}\right]\}) \oplus \mathcal{E}^{\bar{p}}(\Sigma(t)). \tag{25}$$

It follows that if the robot chooses its target configuration $\mathbf{c}$ outside $\mathcal{LQG}^{\bar{p}}(\hat{\mathbf{x}})$, the probability of colliding with an obstacle at any given time $t > 0$ is less than $\bar{p}$, if LQG control is used to control the robot towards $\mathbf{c}$. LQG-Obstacles can be used for navigation in a similar way as LQR-Obstacles (see Fig. 5).

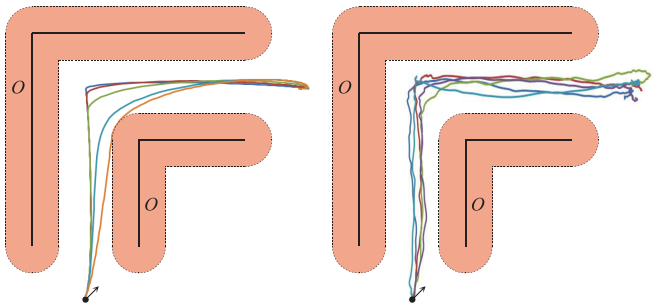


Fig. 5. Controlling the robot (see Section V-B) using LQG-Obstacles in the same environment as Fig. 3. (a) Traces of maximum likelihood executions for $\bar{p} = \{0.01, 0.02, 0.03, 0.05, 1\}$. (b) Traces of five actual executions with synthetic motion and observation noise for $\bar{p} = 0.01$.

C. LQG-Obstacles for Non-Linear Systems

The above derivations only work for linear systems. Let us consider a non-linear Gaussian system of the form:

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}, \mathbf{m}), \qquad \mathbf{m} \sim \mathcal{N}(\mathbf{0}, M), \tag{26}$$
$$\mathbf{z} = h(\mathbf{x}, \mathbf{n}), \qquad \mathbf{n} \sim \mathcal{N}(\mathbf{0}, N). \tag{27}$$

In this case, we can approximate the LQG-Obstacle by linearizing the system. It is convenient to linearize around a steady state $\bar{\mathbf{x}}$ for which $f(\bar{\mathbf{x}}, \mathbf{0}, \mathbf{0}) = \mathbf{0}$. Typically, one chooses $\bar{\mathbf{x}}$ closest to the current state estimate $\hat{\mathbf{x}}$. Linearizing then gives:

$$\dot{\tilde{\mathbf{x}}} = A\tilde{\mathbf{x}} + B\mathbf{u} + \tilde{\mathbf{m}}, \qquad \tilde{\mathbf{m}} \sim \mathcal{N}(\mathbf{0}, VMV^T), \tag{28}$$
$$\tilde{\mathbf{z}} = H\tilde{\mathbf{x}} + \tilde{\mathbf{n}}, \qquad \tilde{\mathbf{n}} \sim \mathcal{N}(\mathbf{0}, WNW^T), \tag{29}$$

where $\tilde{\mathbf{x}} = \mathbf{x} - \bar{\mathbf{x}}$ is the redefined state, $\tilde{\mathbf{z}} = \mathbf{z} - h(\bar{\mathbf{x}}, \mathbf{0})$ is the redefined measurement vector, and $A = \frac{\partial f}{\partial \mathbf{x}}(\bar{\mathbf{x}}, \mathbf{0}, \mathbf{0})$, $B = \frac{\partial f}{\partial \mathbf{u}}(\bar{\mathbf{x}}, \mathbf{0}, \mathbf{0})$, $V = \frac{\partial f}{\partial \mathbf{m}}(\bar{\mathbf{x}}, \mathbf{0}, \mathbf{0})$, $H = \frac{\partial h}{\partial \mathbf{x}}(\bar{\mathbf{x}}, \mathbf{0})$, and $W = \frac{\partial h}{\partial \mathbf{n}}(\bar{\mathbf{x}}, \mathbf{0})$ are the Jacobian matrices of f and h. If the linearized system is *controllable*, we can construct the LQG-Obstacle as above. As the linearized system is only valid in a small region around the linearization point, the models should be continually relinearized to get meaningful control.

V. Implementation and Experimentation

A. Implementation Details

We implemented our approach using a collision-checker capable of performing linear transformations on the geometry [31]. In each time-step of a continuous cycle, we select a target configuration $\mathbf{c} \notin \mathcal{LQG}^{\bar{p}}(\hat{\mathbf{x}})$, and apply the control input $\mathbf{u}$ according to Eq. (19). In our current implementation, we select the configuration $\mathbf{c}$ using a brute-force approach as follows.

Given an explicit representation of $\mathcal{O}$ in the collision-checker, and a finite set $\mathcal{T}$ of candidate target configurations $\mathbf{c}$, we iterate over time $0 < t < \infty$ in small steps Δt, and transform the obstacle $\mathcal{O}$ in the collision-checker to $\mathcal{Q}(t) = (\tilde{C}G(t))^{-1}(\mathcal{O} \oplus \{-\tilde{C}F(t)\left[\begin{smallmatrix}\hat{\mathbf{x}}\\ \dot{\hat{\mathbf{x}}}\end{smallmatrix}\right]\})$ (see Eq. (25)). Then, we iterate over all candidate configurations $\mathbf{c} \in \mathcal{T}$, and use the collision-checker to check whether the ellipse $\mathcal{E}^{\bar{p}}(\Sigma(t))$ centered at $\mathbf{c}$ intersects the transformed obstacle $\mathcal{Q}(t)$. If so, $\mathbf{c}$ is inside the LQG-Obstacle, and is removed from the set $\mathcal{T}$. Obviously, we cannot iterate time t over an infinite domain, but the transformed obstacle $\mathcal{Q}(t)$ converges *exponentially* fast to $\mathcal{O}$ for $t \to \infty$ (this follows from the fact that the LQG controller reaches the target exponentially fast [2]). So, we can safely bound the time domain; in our experiments, we used $0 < t < 4$ in our experiments. In general one would choose a time-bound based on the eigenvalues of matrix $\tilde{A}$, as they determine the precise rate of convergence to the target.

After this, we are left with a reduced set $\mathcal{T}$ of candidate target configurations $\mathbf{c}$ which are outside $\mathcal{LQG}^{\bar{p}}(\hat{\mathbf{x}})$. From this set, we select the most preferable one. In our implementation, the set $\mathcal{T}$ initially consists of the configurations along a guiding path $\pi \in \mathcal{C}$, and the one furthest along the path that remains is chosen.

B. Robot Models

We implemented our approach for two robot models; a linear planar robot whose acceleration can be controlled, and a quadrotor helicopter with non-linear dynamics and observations flying in 3-D. The former was used to generate Figs. 2, 3, 4, and 5 for illustration purposes. The latter is used to report simulation results.

1) Planar Robot with Acceleration Control: The robot is a disc in the plane that is capable of accelerating omni-directionally. Its (linear) dynamics are defined by:

$$\mathbf{x} = \begin{bmatrix}\mathbf{p}\\ \mathbf{v}\end{bmatrix}, \quad \mathbf{u} = \mathbf{a}, \quad A = \begin{bmatrix}0 & I\\ 0 & 0\end{bmatrix}, \quad B = \begin{bmatrix}0\\ I\end{bmatrix}, \quad C = \begin{bmatrix}I & 0\end{bmatrix},$$

where $\mathbf{p} \in \mathbb{R}^2$ is the position, $\mathbf{v} \in \mathbb{R}^2$ the velocity, and $\mathbf{a} \in \mathbb{R}^2$ the acceleration of the robot. The configuration space consist of all positions of the robot (the velocity does not change its geometry in the workspace), so the matrix C as given above projects the state to the robot's configuration.

We used the following settings for the controller and for the stochastic case with noisy motion and partial observation:

$$Q = I, \quad R = I, \quad H = \begin{bmatrix}I & 0\end{bmatrix}, \quad M = 0.01I, \quad N = 0.01I.$$

That is, the robot receives measurements of only its position.

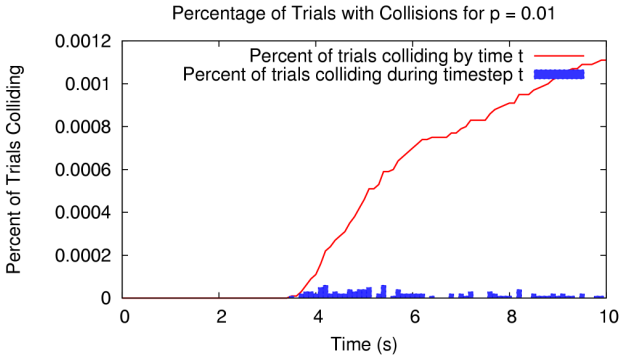


Fig. 6. Probability of collision over time for the LQG-Obstacle with $\bar{p} = 0.01$.

2) Quadrotor Helicopter: Our quadrotor model is based on Ascending Technologies' ResearchPilot. Its 13-D state and 4-D control input are defined by:

$$\mathbf{x} = \begin{bmatrix}\mathbf{p}^T & \mathbf{v}^T & \mathbf{r}^T & \mathbf{w}^T & \varphi\end{bmatrix}^T, \quad \mathbf{u} = \begin{bmatrix}\mathbf{w}^{\star T} & \varphi^\star\end{bmatrix}^T,$$

where $\mathbf{p} \in \mathbb{R}^3$ is the robot's position, $\mathbf{v} \in \mathbb{R}^3$ its velocity, $\mathbf{r} \in \mathbb{R}^3$ its orientation (axis $\mathbf{r}$ and angle $\|\mathbf{r}\|$), $\mathbf{w} \in \mathbb{R}^3$ its angular velocity, and φ the combined force of the rotors. The control input consists of the desired angular velocity $\mathbf{w}^\star$ and force $\varphi^\star$. A low-level on-board controller transforms these into voltages for the motors of each of the rotors. The robot's geometry is modeled by an encapsulating sphere, such that the configuration space consists of only the positions $\mathbf{p} \in \mathbb{R}^3$.

Its non-linear dynamics are modeled after [14], augmented with effects of rotor drag and induced inflow that cause a force in the opposite direction of the velocity [13]:

$$\dot{\mathbf{p}} = \mathbf{v}, \quad \dot{\mathbf{v}} = \begin{bmatrix}0\\0\\-g\end{bmatrix} + (\exp([\mathbf{r}])\begin{bmatrix}0\\0\\\varphi\end{bmatrix} - k_{\mathbf{v}}\mathbf{v})/m,$$
$$\dot{\mathbf{r}} = \mathbf{w} + [\mathbf{r}]\mathbf{w}/2, \quad \dot{\mathbf{w}} = k_{\mathbf{w}}(\mathbf{w}^\star - \mathbf{w}), \quad \dot{\varphi} = k_\varphi(\varphi^\star - \varphi),$$

where $g = 9.8\text{m/s}^2$ is the gravity, m the mass of the robot, and $k_{\mathbf{v}}$, $k_{\mathbf{w}}$, and k_φ scaling constants.

An overhead camera positioned at $\check{\mathbf{p}}$ measures the apparent position $\mathbf{b} \in \mathbb{R}^2$ and radius ρ in the camera image of a ball with radius r that is fixed on top of the quadrotor. In addition, the quadrotor is equipped with an accelerometer, a magnetometer, a rate-gyro, and an altimeter that produce measurements $\mathbf{a} \in \mathbb{R}^3$, $\mathbf{d} \in \mathbb{R}^3$, $\mathbf{g} \in \mathbb{R}^3$, and η, respectively, according to the following non-linear observation model:

$$\mathbf{b} = \begin{bmatrix}(\mathbf{p}_x - \check{\mathbf{p}}_x)\\(\mathbf{p}_y - \check{\mathbf{p}}_y)\end{bmatrix}/(\check{\mathbf{p}}_z - \mathbf{p}_z), \quad \rho = \arcsin(r/\|\mathbf{p} - \check{\mathbf{p}}\|),$$
$$\mathbf{a} = (\begin{bmatrix}0\\0\\\varphi\end{bmatrix} - \exp([\mathbf{r}])^T k_{\mathbf{v}}\mathbf{v})/m, \quad \mathbf{d} = \exp([\mathbf{r}])^T\mathbf{k}_{\mathbf{d}},$$
$$\mathbf{g} = \mathbf{w}, \quad \eta = \mathbf{p}_z,$$

where $\mathbf{k}_{\mathbf{d}} \in \mathbb{R}^3$ is the direction of Earth's magnetic field.

C. Simulation Results

1) Collision Probability Experiment: In the first experiment, we explore the relation between the probability bound parameter $\bar{p}$ of LQG-Obstacles and actual probabilities of collision. Since our LQG-Obstacle formulation is conservative, we expect that the collision probability at a specific

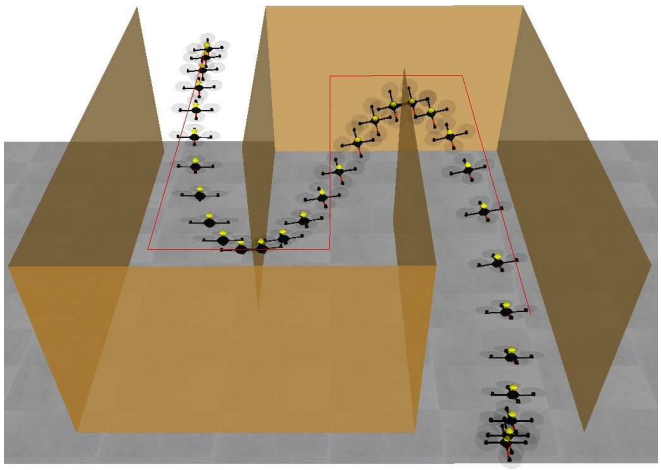
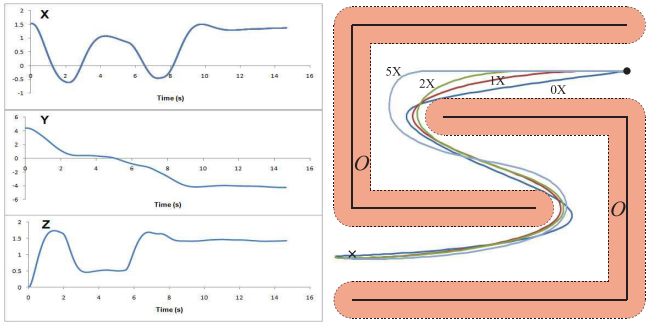


Fig. 7. The *S-Maze* scenario for our quadrotor simulation experiments. The thin red line is the guiding path. The quadrotor is shown at 6Hz. Videos are available at `http://gamma.cs.unc.edu/CA/LQGObs/`.



(a) (b)

Fig. 8. (a) The x, y, and z position over time of the quadrotor in the *3-Window* scenario (see Fig. 1). (b) The xy-projections of the maximum likelihood traces of the quadrotor in the *S-Maze* scenario (see Fig. 7) for 0, 1, 2, and 5$\times$ the realistic amount of uncertainty.

time t will be far lower than our bound, and that the cumulative probability of collision will slowly grow over the duration of the experiment. Given a long enough experiment duration, we suspect this probability will approach one, as even after the robot reaches its goal configuration noise in its motion model will cause it to move unpredictably, eventually bringing the robot into contact with the obstacle.

To test this, we use the robot as described in Section V-B in the scenario of Fig. 4. We select a target configuration $\mathbf{c}$ once on the boundary of the initial LQG-Obstacle of Fig. 4 for $\bar{p} = 0.01$, and control the robot using the LQG-controller towards c for the duration of the experiment with synthetic motion and sensing noise. In Fig. 6, we show the results of this experiment averaging over 100,000 trials. These results confirm that our bound is (very) conservative: the maximal probability of colliding at a specific time t seems to be a factor 100 lower than our bound.

2) Quadrotor Simulation Experiments: To analyze the effectiveness of the LQG-Obstacle technique for navigating the simulated quadrotor we created two experimental scenarios. For both scenarios we used a simulated version of a lab space of approximately 10m$\times$5m. A simulated overhead camera which refreshes at 30Hz is used to augment the quadrotor's on-board sensing to allow localization in the environment. It is assumed that the location of all obstacles are known in advance. For both scenarios we set $\bar{p} = 0.03$ and used as realistic model parameters and levels of motion and sensor uncertainty as possible.

In the *3-Window* scenario (see Fig. 1), the quadrotor must navigate from the east side of the room to the west side. In between there is a series of three small windows the quadrotor must pass through at various heights and positions. A simple guiding path is given consisting of several short, straight segments through the center of each window. In the *S-Maze* scenario (see Fig. 7), the quadrotor starts on the floor in the southeast corner of a room and is given a goal configuration in the air in the northwest corner. Between these two positions are several walls creating an S-shaped corridor the quadrotor must navigate through. A simple guiding path is given consisting of four straight lines through the center of the corridor.

In both scenarios, the quadrotor is able to smoothly and consistently navigate to its goal position without colliding into the walls. The graphs in Fig. 8(a) show the 3-D motion of the quadrotor in the *3-Window* scenario. Videos of both scenarios are available at `http://gamma.cs.unc.edu/CA/LQGObs/`.

To demonstrate the effect of motion and sensing noise we ran the *S-Maze* scenario with various levels of noise. First with no simulated noise, second with realistic levels of noise, and then with 2$\times$ and 5$\times$ more noise than the realistic amount used in other experiments. Beyond 5$\times$ the quadrotor failed to reach its goal, and instead hovered near the entrance of the maze. Fig. 8(b) shows an xy-projection of the maximum likelihood path for each of these runs of increasing noise. By comparing the no-noise run to the standard noise run the effect of the LQG-Obstacle is apparent. The quadrotor takes a clearly wider turn around the first bend to avoid coming too close to the wall. At higher levels of noise, the quadrotor takes a very conservative path, staying close the center of the corridor for much of the run. While the constraints of the flight dynamics reduce the variation in paths during the second curve, the higher noise runs still stay further away from walls. In general, planning for more noise allowed smoother, gentler paths.

3) Timing Results: A benefit of LQG-Obstacles is the ability to run fast enough for feedback control. For the quadrotor scenarios, the computation can not take more than 33.3ms without frames from the camera being lost. Here we report the execution time for planning on an 3.2GHz Intel Core i7. Table I shows the time taken by the LQG-Obstacle feedback planner for the three scenarios discussed in this paper. In all cases the computation is time faster than the control frequency of 30Hz. The number of considered target configurations – which in our case is proportional to the length of the guiding path – is the main factor determining the computation time.

TABLE I

COMPUTATION TIME FOR THREE DIFFERENT SCENARIOS.

Robot	Scenario	Computation Time
Planar	*L-Corridor*	9.8ms (102Hz)
Quadrotor	*S-Maze*	21.4ms (47Hz)
Quadrotor	*3-Window*	24.8ms (40Hz)

VI. CONCLUSION, LIMITATIONS, AND FUTURE WORK

In this paper, we have introduced the new concept of LQG-Obstacles for combining LQG feedback control with collision avoidance. We have shown using simulations that our approach can be used to smoothly and safely fly a quadrotor helicopter with motion and sensing uncertainty through an environment with obstacles and narrow corridors. We are currently working to implement our approach on a real quadrotor.

Our approach has a number of limitations. First, it requires the geometry of the obstacles in the configuration space of the robot to be given explicitly. While in theory our approach works for any robot, in practice its applicability is limited to robots with simple geometry, such as mobile robots that can be modeled as a disc or a sphere. Also, our approach works for non-linear systems only if the linearized dynamics are controllable. For the quadrotor, we linearized about its nearest steady state, but for car-like robots or differential-drives one has to choose the linearization point more carefully, as in these cases linearizing about a steady state results in non-controllable dynamics. Further, our current implementation lets the robot select its target configuration along a guiding path. This is neither the optimal way, nor the only way to use LQG-Obstacles. Investigating alternative strategies for selecting control objectives is subject of ongoing work.

There are a few relatively straightforward extensions to our method we did not discuss in this paper. Firstly, our approach can handle constraints on the control input. These translate to constraints on the target configuration through the control policy, and can be included in the LQG-Obstacle. Also, our approach can be extended to work for moving obstacles, by replacing $\mathcal{O}$ by $\mathcal{O}(t)$ in the derivation. Extensions that require further study include accounting for uncertainty in the geometry and motion of obstacles, and on-board (local) sensing of obstacles. Our approach may also extend to reciprocal collision avoidance [29] for multiple robots.

REFERENCES

[1] D. Althoff, M. Althoff, D. Wollherr, M. Buss. Probabilistic collision state checker for crowded environments. *IEEE Int. Conf. on Robotics and Automation*, 2010.

[2] M. Athans, P. Falb. *Optimal Control: An Introduction to the Theory and Its Applications*. Dover Publications, 2006.

[3] P. Fiorini, Z. Shiller. Motion planning in dynamic environments using velocity obstacles. *Int. Journal of Robotics Research* 17(7):760, 1998.

[4] D. Fox, W. Burgard, S. Thrun. The dynamic window approach to collision avoidance. *IEEE Robotics and Automation Magazine* 4:23–33, 1997.

[5] T. Fraichard, H. Asama. Inevitable collision states - a step towards safer robots? *Advanced Robotics* 18(10):10011024, 2004.

[6] C. Fulgenzi, A. Spalanzani, C. Laugier. Dynamic obstacle avoidance in uncertain environment combining PVOs and occupancy grid. *IEEE Int. Conf. on Robotics and Automation*, 2007.

[7] A. Gelb. *Applied optimal estimation*. The Analytic Science Corporation, 1974.

[8] V. Huynh, N. Roy. icLQG: combining local and global optimization for control in information space. *IEEE Int. Conf. on Robotics and Automation*, 2009.

[9] S. LaValle, J. Kuffner. Randomized kinodynamic planning. *Int. Journal on Robotics Research* 20(5):378–400, 2001.

[10] L. Kaelbling, M. Littman, A. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence* 101(1-2):99–134, 1998.

[11] V. Kumar, N. Michael. Opportunities and challenges with autonomous micro aerial vehicles. *Int. Symp. on Robotics Research*, 2011.

[12] H. Kurniawati, D. Hsu, W. Lee. SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces. *Robotics: Science and Systems*, 2008.

[13] P. Martin, E. Salaün. The true role of accelerometer feedback in quadrotor control. *IEEE Int. Conf. on Robotics and Automation*, 2010.

[14] N. Michael, D. Mellinger, Q. Lindsey, V. Kumar. The GRASP multiple mirco-UAV test bed: experimental evaluation of multirobot aerial control algorithms. *IEEE Robotics and Automation Magazine* 17(3):56-65, 2010.

[15] L. Kavraki, P. Svestka, J.-C. Latombe, M. Overmars. Probabilistic roadmaps for path planning in high dimensional configuration spaces. *IEEE Trans. on Robotics and Automation* 12:4(566–580), 1996.

[16] L. Lu, C. Pearce. On the square-root method for continuous-time algebraic Riccati equations. *Journal of the Australian Mathematical Society*, Series B, 40:459–468, 1999.

[17] E. Owen, L. Montano. Motion planning in dynamic environments using the velocity space. *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2005.

[18] C. Papadimitriou, J. Tsisiklis. The complexity of Markov decision processes. *Mathematics of Operations Research*, 12(3):441–450, 1987.

[19] S. Petti, T. Fraichard. Safe motion planning in dynamic environments. *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2005.

[20] J. Porta, N. Vlassis, M. Spaan, P. Poupart. Point-based value iteration for continuous POMDPs. *Journal of Machine Learning Research* 7:2329–2367, 2006.

[21] R. Platt, R. Tedrake, L. Kaelbling, T. Lozano-Perez. Belief space planning assuming maximum likelihood observations. *Robotics: Science and Systems*, 2010.

[22] S. Prentice, N. Roy. The belief roadmap: Efficient planning in belief space by factoring the covariance. *Int. J. of Robotics Research* 28(1112):1448-1465, 2009.

[23] J. Rawlings. Tutorial overview of Model Predictive Control. *IEEE Control Systems Magazine* 20(3):38–52, 2000.

[24] J. Snape, J. van den Berg, S. Guy, D. Manocha. Independent navigation of multiple robots with Hybrid Reciprocal Velocity Obstacles. *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2009.

[25] R. Tedrake. I. Manchester, M. Tobenkin, J. Roberts. LQR-trees: Feedback motion planning via sums-of-squares verification. *Int. J. on Robotics Research* 29(8):1038-1052, 2010.

[26] S. Thrun, W. Burgard, D. Fox. *Probabilistic Robotics*, MIT Press, 2005.

[27] E. Todorov, W. Li. A generalized iterative LQG method for locally-optimal feedback control of constrained nonlinear stochastic systems. *American Control Conference*, 2005.

[28] J. van den Berg, P. Abbeel, K. Goldberg. LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information. *Int. J. of Robotics Research* 30(7):895-913, 2011.

[29] J. van den Berg, J. Snape, S. Guy, D. Manocha. Reciprocal collision avoidance with acceleration-velocity obstacles. *IEEE Int. Conf. on Robotics and Automation*, 2011.

[30] J. van den Berg, S. Patil, R. Alterovitz. Motion planning under uncertainty using differential dynamic programming in belief space. *Int. Symp. on Robotics Research*, 2011.

[31] G. van den Bergen. *Collision detection in interactive 3D environments*. Morgan Kaufmann Publishers, 2004.

[32] D. Wilkie, J. van den Berg, and D. Manocha. Generalized velocity obstacles. *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2009.