Periodic-node Graph-based Framework for Stochastic Control of Non-point-stabilizable Systems

Ali-akbar Agha-mohammadi, Saurav Agarwal, and Suman Chakravorty

Abstract

This paper presents a strategy for stochastic control of small aerial vehicles under uncertainty using graph-based methods. In planning with graph-based methods, such as the Probabilistic Roadmap Method (PRM) in state space or the Information RoadMaps (IRM) in information-state (belief) space, the local planners (along the edges) are responsible to drive the state/belief to the final node of the edge. However, for aerial vehicles with minimum velocity constraints, driving the system belief to a sampled belief is a challenge. In this paper, we propose a novel method based on periodic controllers, in which instead of stabilizing the belief to a predefined probability distribution, the belief is stabilized to an orbit (periodic path) of probability distributions. Choosing nodes along these orbits, the node reachability in belief space is achieved and we can form a graph in belief space that can handle higher-order-dynamics or non-stoppable systems (whose velocity cannot be zero), such as fixed-wing aircraft. The proposed method takes obstacles into account and provides a query-independent graph, since its edge costs are independent of each other, thus it satisfies the principle of optimality. Therefore, dynamic programming can be utilized to compute the best feedback on the graph. We demonstrate the method's performance on a unicycle robot and a six degree of freedom small aerial vehicle.

I. Introduction

This paper is concerned with the stochastic control problem for two classes of systems: *(i)* systems with dynamics, i.e., systems, whose state is composed of position and its higher order derivatives such as velocity and acceleration, and/or *(ii)* systems with kinodynamical constraints, in particular, systems, whose velocity cannot fall below a certain threshold (referred to as non-stoppable systems in this paper). For example, consider a control problem where the system state is composed of the position and velocity $(x, \dot{x})$ of an object. Stabilizing this system to a state $(x = a, \dot{x} = b)$ where $b \neq 0$ is not possible since to stabilize the x part to a, the $\dot{x}$ must go to zero. As an example for non-stoppable systems, consider a system, whose state only consists of position x, but it has constraints on its velocity $\dot{x} > b > 0$. All fixed-wing aircraft fall into this category as their velocity cannot fall below some threshold to maintain the lift requirement. Thus, stabilizing such systems to a fixed state is a challenge. This challenge gets even more difficult when this stabilization has to be achieved under uncertainty. In this paper, we propose a framework that circumvents the need for point stabilization in graph-based (roadmap-based) methods by means of stabilization to suitably designed periodic maneuvers.

Motion planning under uncertainty (MPUU) is an instance of the problem of sequential decision making under uncertainty. Considering the uncertainty in an object's motion the problem can be framed as a stochastic control with perfect state information. This problem has received a lot of attention and methods such as [6] have investigated its application on small aerial vehicles under stochastic wind. In the presence of uncertainty in sensory readings the state of the system is no longer available for decision making. In such a situation, a state estimation module can provide a probability distribution (referred to as information-state or belief) over all possible states of the system, and therefore decision making has to be performed in belief space. Planning in belief space in its most general form is formulated as a Partially Observable Markov Decision Process (POMDP) problem [7], [17]. However, in general solving POMDPs in continuous state, control, and observation spaces, where many robotic problems reside, is a formidable challenge.

Sampling-based motion planning methods have shown great success in dealing with many motion planning problems in complex environments and are divided into two main classes: *(i)* roadmap-based (graph-based) methods such as the Probabilistic Roadmap Method (PRM) and its variants [5], [18], [19] and *(ii)* tree-based methods such as methods in [15], [18], [21]. In deterministic setting, tree-based methods are usually single-query, i.e., their solution is valid for a given initial point whereas roadmap-based methods are mainly multi-query, i.e., the generated roadmap structure is independent of the initial point. In this sense roadmap-based methods are a suitable choice for extension to belief space because the solution of POMDP is a feedback over whole belief space and it does not depend on the initial belief. Accordingly, restricting the attention to a representative graph (roadmap) in the space, the feedback can be defined as a mapping from its nodes to its edges.

Similar to motion planning in state space, in belief space motion planning, the basic motion tasks can be defined as: *point-to-point motion*, which deals with driving the belief of the moving object from a given belief to another given belief, and *trajectory following*, which deals with following a trajectory in belief space. Depending on the kinematics/dynamics of the system, these tasks might be very challenging in the state space. However, they often are more challenging in belief space even for simple kinematics/dynamics. To construct a query-independent roadmap in state/belief space, point-to-point motion in state/belief space is required. Feedback-based Information RoadMap (FIRM) [2], [4] extends graph-based methods

A. Agha-mohammadi is with the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA, 02139. S. Agarwal and S. Chakravorty are with the Dept. of Aerospace, Texas A&M University, College Station, TX 77843, USA. Emails: aliagha@mit.edu, sauravag@tamu.edu, schakrav@tamu.edu.

to belief space by embedding the point-to-point motion behavior in belief space using belief stabilizers (stationary feedback controllers), which was a missing behavior in pioneering work such as [16], [23], [27] that exploit the graph structure in belief space planning.

As a result of embedding the point-to-point motion behavior in belief space, FIRM generates a graph in belief space that is query-independent and only needs to be constructed once offline. Establishing a connection between its solution and the original POMDP [4], it is shown that FIRM is probabilistically complete [3]. In [4] first FIRM is presented as an abstract framework for graph-based planning in belief space and then Stationary Linear Quadratic Gaussian-FIRM (SLQG-FIRM) is presented as a concrete instantiation of the abstract FIRM framework. However, SLQG-FIRM is limited to the systems that are stabilizable to stationary fixed points (with zero velocity) in the state space. This excludes the class of systems we consider in this paper.

The main contributions of this paper are:

- Proposing a graph-based solution for controlling small aerial vehicles in the presence of uncertainty and constraints. We accomplish this goal by proposing a concrete instantiation of the FIRM framework that can handle non-stoppable systems (i.e., class of dynamical systems that are not stabilizable to a point with zero-velocity), such as fixed-wing aircraft.
- Accordingly, transforming the intractable constrained POMDP to a tractable dynamic programming over a graph corresponding to non-stoppable systems.
- Designing the periodic-node PRM in state space.
- Investigating the cyclostationary behavior of the belief under Periodic Linear Quadratic Gaussian (PLQG) controllers and designing a belief stabilizer for non-stoppable systems.

This paper is organized as follows. We start by introducing the concept of periodic-node graph in state space, whose nodes lie on periodic trajectories referred to as orbit. In Section III, we review the problem of stochastic optimal control with imperfect observations. Section IV constructs the abstract FIRM framework based on the underlying periodic-node graph. In this section, we show how constraints are incorporated in the construction phase of the planner. Then, in Section V we analyze the behavior of PLQG controllers as belief stabilizers and accordingly we propose an approach to characterize and select the reachable regions in belief space under PLQG controllers. As a result we extend the periodic-node PRM from state space to a corresponding graph in belief space. We provide algorithms for offline construction of this graph and online (re)planning with this graph. Finally, in Section VI, we demonstrate the performance of the proposed method on a planar unicycle model with minimum allowable velocity and on a simplified 6DoF aerial vehicle model.

II. Periodic-Node PRM

An implicit assumption in graph-based methods such as PRM [19] is that on every edge there exists a controller to drive the robot from the start node of the edge to the end node of the edge or to an ϵ-neighborhood of the end node, for a sufficiently small $\epsilon > 0$. For a linearly controllable robot, a linear controller can locally track a PRM edge and drive the robot to its endpoint node. Obviously, controlling non-stoppable robots on a PRM roadmap is a challenge, since they have constraints on their controls and cannot reduce their velocity below a specific threshold u_{min}, and hence, stabilization is not feasible for them. This task becomes more challenging if the system is also nonholonomic. In a nonholonomic robot such as a unicycle, the linearized model at any point is not controllable, and hence, a linear controller cannot stabilize the robot to the PRM nodes. Consider the discrete unicycle model:

$$x_{k+1} = f(x_k, u_k, w_k) = \begin{pmatrix} \mathsf{x}_k + (V_k + n_v)\delta t \cos\theta_k \\ \mathsf{y}_k + (V_k + n_v)\delta t \sin\theta_k \\ \theta_k + (\omega_k + n_\omega)\delta t \end{pmatrix}, \quad w_k \sim \mathcal{N}(0, \mathbf{Q}_k) \tag{1}$$

where $x_k = (\mathsf{x}_k, \mathsf{y}_k, \theta_k)^T$ describes the robot state, in which $(\mathsf{x}_k, \mathsf{y}_k)^T$ is the 2D position of the robot and θ_k is the heading angle of the robot, at time step k. The vector $u_k = (V_k, \omega_k)^T$ is the control vector consisting of linear velocity V_k and angular velocity ω_k. The motion noise vector is denoted by $w_k = (n_v, n_\omega)^T$. Linearizing this system about the point (node) $\mathbf{v} = (\mathsf{x}^p, \mathsf{y}^p, \theta^p)$, nominal control $u^p = (V^p, \omega^p)$, and zero noise, we get:

$$x_{k+1} = f(x_k, u_k, w_k) \approx f(\mathbf{v}, u^p, 0) + \mathbf{A}(x_k - \mathbf{v}) + \mathbf{B}(u_k - u^p) + \mathbf{G}w_k \tag{2}$$

where $\mathbf{A} = \frac{\partial f}{\partial x}(\mathbf{v}, u^p, 0)$, $\mathbf{B} = \frac{\partial f}{\partial u}(\mathbf{v}, u^p, 0)$,, $\mathbf{G} = \frac{\partial f}{\partial w}(\mathbf{v}, u^p, 0)$. Checking the rank of the controllability matrix of the linearized system, we get: $rank([\mathbf{B}, \mathbf{AB}, \mathbf{A}^2\mathbf{B}]) = 2 + \mathbb{I}(V^p > 0)$, where $\mathbb{I}$ is the indicator function, which is one if $V^p > 0$ and is zero, otherwise. Therefore, if the nominal control is zero, $u^p = (V^p, \omega^p)^T = (0, 0)^T$, which is the case when we stabilize the robot to a PRM node, the resulting linear system is not controllable, since $rank([\mathbf{B}, \mathbf{AB}, \mathbf{A}^2\mathbf{B}]) = 2 < 3$. Thus, a linear controller cannot stabilize the unicycle to a PRM node. Moreover, based on the necessary condition in Brockett's paper [11], even a smooth time-invariant nonlinear control law cannot drive the unicycle to a PRM node, and the stabilizing controller has to be either discontinuous and/or time-varying.

On roadmaps in belief space, the situation is even more complicated, since the controller has to drive the probability distribution over the state to the ϵ-neighborhood of a belief node in belief space. Again, if the linearized system in (2) is controllable, using a linear stochastic controller such as the stationary LQG controller, one can drive the robot belief to the belief node [4]. However, if the system is non-stoppable and/or its linearized model is not controllable, the belief stabilization, if possible, is much more difficult than state stabilization.

A. *Periodic-node PRM*

In this paper, we circumvent the problem of stabilization to graph nodes by designing a variant of PRM, referred to as Periodic-Node PRM (PNPRM). Although there are different ways to address this problem in state space, the critical property of PNPRM is that it can be extended to belief space to form a graph whose nodes are beliefs that are reachable without a point-stabilization process. Let us denote the motion model with $x_{k+1} = f(x_k, u_k, w_k)$, where state, control, and process noise at the k-th time step are denoted by x_k, u_k, and w_k, respectively.

Similar to traditional PRM, PNPRM also consists of nodes and edges. However, in PNPRM, the nodes lie on small T-periodic trajectories (trajectories with period T) in the state space, referred to as orbits. Each orbit satisfies the control constraints and non-holonomic constraints of the moving robot. To construct a PNPRM, we first sample a set of orbits in the state space, and then on each orbit, a number of state nodes are selected. Let us denote the j-th orbit trajectory by $O^j := (x_k^{p^j}, u_k^{p^j})_{k\geq 0}$, where $x_{k+1}^{p^j} = f(x_k^{p^j}, u_k^{p^j}, 0)$, $x_{k+T}^{p^j} = x_k^{p^j}$, and $u_{k+T}^{p^j} = u_k^{p^j}$. The set of PNPRM nodes that are chosen on O^j is denoted by $\mathbb{V}^j = \{\mathbf{v}_1^j, \mathbf{v}_2^j, \cdots, \mathbf{v}_m^j\}$ where $\mathbf{v}_\alpha^j = x_{k_\alpha}^{p^j}$ for some $k_\alpha \in \{1, \cdots, T\}$. Edges in PNPRM do not connect nodes to nodes, but they connect orbits to orbits in a way that respects all the control constraints and nonholonomic constraints. Thus, the (i, j)-th edge denoted by $\mathbf{e}^{ij}$ connects O^i to O^j.

As a result, a node $\mathbf{v}_\alpha^i$ is connected to the node $\mathbf{v}_\gamma^j$ through concatenation of three path segments: *i)* the first segment is a part of O^i that connects $\mathbf{v}_\alpha^i$ to the starting point of $\mathbf{e}^{ij}$. This part is called *pre-edge* and is denoted by $\mathbf{e}^{i_\alpha j}$, *ii)* the second segment is the edge $\mathbf{e}^{ij}$ itself that connects O^i to O^j, and *iii)* the third segment is a part of O^j that connects the ending point of $\mathbf{e}^{ij}$ to $\mathbf{v}_\gamma^j$. This part is called *post-edge* and is denoted by $\mathbf{e}^{ij_\gamma}$.

One form of constructing orbits is based on circular periodic trajectories, where the edges are the lines that are tangent to the orbits. Figure 1 shows a simple PNPRM with three orbits O^i, O^r, and O^j. On each orbit four nodes are selected which are drawn (dots) with different colors. Edges $\mathbf{e}^{ij}$ and $\mathbf{e}^{rj}$ connect the corresponding orbits.

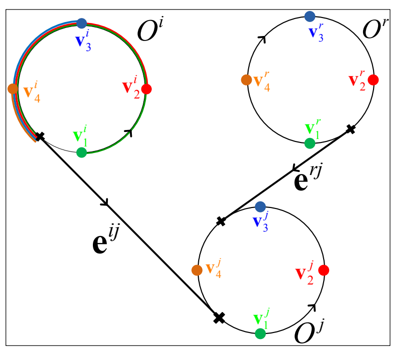


Fig. 1. A simple PNPRM with three orbits, twelve nodes, and two edges.

III. Planning in Information Space

Partially Observable Markov Decision Processes (POMDPs) are the most general formulation for motion planning problems under motion and sensing uncertainties. Note that in this paper, the environment map is assumed to be known. The solution of the POMDP problem is an optimal feedback law (mapping) π, which maps the information (belief) space to the control space. Let us denote the state, control and observation at time step k by x_k, u_k, and z_k, respectively, which belong to spaces $\mathbb{X}$, $\mathbb{U}$, and $\mathbb{Z}$, respectively. The belief in stochastic setting is defined as the probability distribution function (pdf) of the system state conditioned on the obtained measurements and applied controls up to the k-th time step, i.e., $b_k := p(x_k | z_{0:k}; u_{0:k-1})$ and $\mathbb{B}$ denotes the belief space, containing all possible beliefs. Note that $z_{0:k} = \{z_1, z_2, \cdots, z_k\}$ and $u_{0:k-1} = \{u_1, u_2, \cdots, u_{k-1}\}$. It is well known that the POMDP problem can be posed as a Markov Decision Process (MDP) in belief space [8], [26], whose solution π is computed by solving the following Dynamic Programming (DP) equation:

$$J(b) = \min_u \{c(b,u) + \int_{\mathbb{B}} p(b'|b,u) J(b') db'\}, \quad \forall b \in \mathbb{B} \tag{3a}$$

$$\pi(b) = \arg\min_u \{c(b,u) + \int_{\mathbb{B}} p(b'|b,u) J(b') db'\}, \quad \forall b \in \mathbb{B} \tag{3b}$$

where $J(\cdot) : \mathbb{B} \to \mathbb{R}$ is the optimal cost-to-go function, $p(b'|b,u)$ is the belief transition pdf under control u, and $c(b,u)$ is the one-step cost of taking control u at belief b.

IV. FIRM FRAMEWORK BASED ON PNPRM

It is well known that the above DP equation is exceedingly difficult to solve since it is defined over an infinite-dimensional belief space. In this section, inspired by sampling-based methods, we build a graph in belief space by sampling beliefs from belief space and connecting them to each other. Hence we reduce the intractable DP in 3 to a tractable DP over this graph.

Graph nodes: Let us denote the nodes and edges of the underlying PNPRM by $\mathcal{V} = \cup_{j=1}^{n}\mathbb{V}^j = \cup_{j=1}^{n}\{\mathbf{v}_\alpha^j\}_{\alpha=1}^{m}$ and $\mathcal{E} = \{\mathbf{e}_{ij}\}$, respectively. Corresponding to each PNPRM node $\mathbf{v}_\alpha^i$, we have a unique belief $\grave{b}_\alpha^j$ whose reachability can be guaranteed utilizing appropriate feedback controllers. A concrete example of designing such a controller and computing $\grave{b}_\alpha^j$ will be provided in Section V. We define the j-th graph node in belief space (or FIRM node) B^j as a neighborhood around $\grave{b}_\alpha^j$; i.e., $B_\alpha^j = \{b : \|b - \grave{b}^j\| \leq \epsilon\}$. The set of FIRM nodes that correspond to the i-th orbit is denoted by $\mathbb{V}^i = \{B_\alpha^i\}_{\alpha=1}^{m}$ and the set of all FIRM nodes is $\mathbb{V} = \cup_{i=1}^{n}\mathbb{V}^i$. Figure 2 shows an example set of Gaussian $\grave{b}^j$'s corresponding to the PNPRM nodes in Fig. 1. In Gaussian case each belief b is characterized by its mean $\hat{x}^+$ and covariance P, denoted by $b \equiv (\hat{x}^+, P)$. In Fig. 2, the mean part of $\grave{b}^j$'s is assumed to coincide with the underlying PNPRM node and the covariance part is shown by its 3σ ellipse. Also FIRM node B_2^j (neighborhood of $\grave{b}_2^j$) is shown.

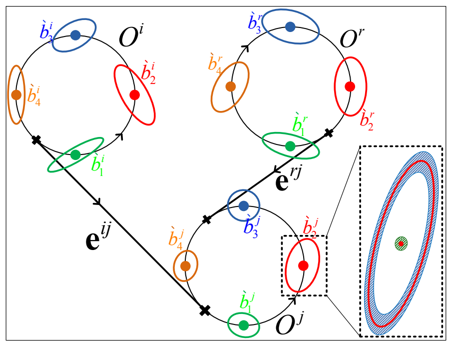


Fig. 2. $\grave{b}_\alpha^l = (\mathbf{v}_\alpha^l, \check{P}_{k_\alpha}^l)$ is the center of belief nodes corresponding to the nodes shown in Fig. 1, where $\check{P}_{k_\alpha}^l$'s are shown by their 3σ-ellipse. As an example of a FIRM node, the magnified version of B_2^j, which is a small neighborhood centered at $\grave{b}_2^j$, is shown in the dotted box, where the blue shaded region depicts the covariance neighborhood and green shaded region depicts the mean neighborhood.

Graph edges: Each graph edge in belief space is a local feedback controller $\mu(\cdot) : \mathbb{B} \to \mathbb{U}$. The role of (i_α, j)-th local controller, denoted by $\mu^{\alpha,ij}$, is to take a belief from the FIRM node B_α^i to a FIRM node on orbit O^j, i.e., to $\cup_\gamma B_\gamma^j$. Thus, we define $\mathcal{T}^{\alpha,ij} := \min\{k \geq 0, b_k \in \cup_\gamma B_\gamma^j | b_0 = \grave{b}_\alpha^j, \mu^{\alpha,ij}\} \in [0, \infty]$ as the stopping time of the controller $\mu^{\alpha,ij}$. The stopping time is a random variable that defines the time it takes for the controller to drive the belief from the initial node to the target orbit. Also, let the $\mathbb{P}(A|b,\mu)$ be the probability of reaching set A in finite time under the local controller μ starting from belief b. Therefore, for a local controller $\mu^{\alpha,ij}$ to act as a graph edge, it has to satisfy $\mathbb{P}(\cup_\gamma B_\gamma^j | \grave{b}_\alpha^i, \mu^{\alpha,ij}) = \Pr(\mathcal{T}^{\alpha,ij} < \infty) = 1$ in the absence of obstacles. In other words, in a constraint-free environment, the feedback controller $\mu^{\alpha,ij}(\cdot)$ has to drive the system's belief from B_α^i into a $B \in \mathbb{V}^j$ in finite time with probability one.

In this section, it is assumed that a set of edges (local controllers) that satisfy the mentioned reachability property is given. In Section V we show that the above property can be accomplished using periodic LQG controller for the class of non-stoppable/nonholonomic systems, such as aircraft. Accordingly, we provide concrete algorithms to construct local controllers and their corresponding reachable nodes.

Graph in belief space: Formally, we define the constructed graph as $G = (\mathbb{V}, \mathbb{M})$ with the set of nodes $\mathbb{V} = \{B_\alpha^j\}$ and the set of edges $\mathbb{M} = \{\mu^{\alpha,ij}\}$. The set of edges available (i.e., outgoing) at FIRM node B_α^i is denoted by $\mathbb{M}(i,\alpha) := \{\mu^{\alpha,ij} \in \mathbb{M} | \exists \mathbf{e}_{ij} \in \mathcal{E}\}$. It is worth noting that the planning is still performed over continuous state, control, and observation spaces and we do not discretize any of those.

Graph transition cost and probabilities: We generalize the one-step transition costs $c(b,u)$ and probabilities to the cost of taking a controller in a graph node and its corresponding transition probabilities along the graph edges:

$$C^g(B_\alpha^i, \mu^{\alpha,ij}) := \sum_{k=0}^{\mathcal{T}^{\alpha,ij}} c(b_k, \mu^{\alpha,ij}(b_k) | b_0 = \grave{b}_\alpha^i) \approx \sum_{k=0}^{\mathcal{T}^{\alpha,ij}} c(b_k, \mu^{\alpha,ij}(b_k) | b_0 = b), \quad \forall b \in B_\alpha^i \tag{4a}$$

$$\mathbb{P}^g(B_\gamma^j | S_\alpha^i, \mu^{\alpha,ij}) := \mathbb{P}(B_\gamma^j | \grave{b}_\alpha^i, \mu^{\alpha,ij}) \approx \mathbb{P}(B_\gamma^j | b, \mu^{\alpha,ij}), \quad \forall b \in B_\alpha^i \tag{4b}$$

The "piecewise constant approximation" in (4) is an arbitrarily good approximation for sufficiently small B_α^i and smooth cost function and transition probabilities.

Graph policy: Graph policy $\pi^g : \mathbb{V} \to \mathbb{M}$ is a function that returns a local controller for any given node of the graph. We denote the space of all graph policies by Π^g.

Graph cost-to-go: To choose the best graph policy, we define the graph cost-to-go J^g from every graph node. Let $B_{\overline{k}}$ be the $\overline{k}$-th FIRM node visited along the plan. Then, we can formally define the cost-to-go from any node $B_0 \in \mathbb{V}$ as:

$$
\begin{aligned}
J^g(B_0;\pi^g) &= \sum_{\overline{k}=0}^{\infty} \mathbb{E}\left[C^g(B_{\overline{k}}, \pi^g(B_{\overline{k}}))\right] \\
s.t. \quad & B_{\overline{k+1}} \sim \mathbb{P}^g(B_{\overline{k+1}}|B_{\overline{k}}, \pi^g(B_{\overline{k}}))
\end{aligned} \tag{5}
$$

Accordingly, the MDP defined on the graph is as follows:

$$
\begin{aligned}
\pi^{g^*} &= \arg\min_{\Pi^g} \sum_{\overline{k}=0}^{\infty} \mathbb{E}\left[C^g(B_{\overline{k}}, \pi^g(B_{\overline{k}}))\right] \\
s.t. \quad & B_{\overline{k+1}} \sim \mathbb{P}^g(B_{\overline{k+1}}|B_{\overline{k}}, \pi^g(B_{\overline{k}}))
\end{aligned} \tag{6}
$$

Obstacle-free graph DP: Since the graph MDP is defined on a finite number of FIRM nodes, we can form a tractable Dynamic Programming (DP) to find the optimal graph policy:

$$
J^g(B^i_\alpha) = \min_{\mu^{\alpha,ij}\in\mathbb{M}(i,\alpha)} C^g(B^i_\alpha, \mu^{\alpha,ij}) + \sum_{\gamma=1}^{m} J^g(B^j_\gamma)\mathbb{P}^g(B^j_\gamma|B^i_\alpha, \mu^{\alpha,ij}),\ \forall \alpha, i, j \tag{7a}
$$

$$
\pi^g(B^i_\alpha) = \arg\min_{\mu^{\alpha,ij}\in\mathbb{M}(i,\alpha)} C^g(B^i_\alpha, \mu^{\alpha,ij}) + \sum_{\gamma=1}^{m} J^g(B^j_\gamma)\mathbb{P}^g(B^j_\gamma|B^i_\alpha, \mu^{\alpha,ij}),\ \forall \alpha, i, j \tag{7b}
$$

where $J^g(\cdot) := \min_{\pi^g} J(\cdot;\pi^g)$ is the optimal cost-to-go.

Incorporating obstacles into the planning: In the presence of obstacles, we cannot assure that the local controller $\mu^{\alpha,ij}(\cdot)$ can drive any $b \in B^i_\alpha$ into $\cup_\gamma B^j_\gamma$ with probability one. Instead, we specify the failure probabilities that the robot collides with an obstacle. Let us denote the failure set on $\mathbb{X}$ by F (i.e., $F = \mathbb{X} - \mathbb{X}_{free}$). Let $\mathbb{P}(F|B^i_\alpha, \mu^{\alpha,ij}) := \mathbb{P}(F|\grave{b}^i, \mu)$ denote the probability of hitting the failure set under local controller $\mu^{\alpha,ij}$ starting from B^i_α. Similarly, we generalize the cost-to-go function by defining $J^g(F)$ as a user-defined suitably high cost for hitting obstacles. Therefore, we can modify (7) to incorporate obstacles in the state space as follows:

$$
\begin{aligned}
J^g(B^i_\alpha) = &\min_{\mu^{\alpha,ij}\in\mathbb{M}(i,\alpha)} C^g(B^i_\alpha, \mu^{\alpha,ij}) + J^g(F)\mathbb{P}^g(F|B^i_\alpha, \mu^{\alpha,ij}) \\
&+ \sum_{\gamma=1}^{m} J^g(B^j_\gamma)\mathbb{P}^g(B^j_\gamma|B^i_\alpha, \mu^{\alpha,ij}),\ \forall \alpha, i, j
\end{aligned} \tag{8a}
$$

$$
\begin{aligned}
\pi^g(B^i_\alpha) = &\arg\min_{\mu^{\alpha,ij}\in\mathbb{M}(i,\alpha)} C^g(B^i_\alpha, \mu^{\alpha,ij}) + J^g(F)\mathbb{P}^g(F|B^i_\alpha, \mu^{\alpha,ij}) \\
&+ \sum_{\gamma=1}^{m} J^g(B^j_\gamma)\mathbb{P}^g(B^j_\gamma|B^i_\alpha, \mu^{\alpha,ij}),\ \forall \alpha, i, j
\end{aligned} \tag{8b}
$$

Thus, all that is required to solve the above DP equation are the values of the costs $C^g(B^i_\alpha, \mu^{\alpha,ij})$ and transition probability functions $\mathbb{P}^g(\cdot|B^i_\alpha, \mu^{\alpha,ij})$, which are discussed in Section V.

Overall policy π: The overall feedback π is generated by combining the policy π^g on the graph and the local controllers $\mu^{\alpha,ij}$s. However, this combination leads to a non-Markov policy. More rigorously, the resulting policy is a semi-Markov policy [25]. In other words, in addition to the current belief the generated action depends on the last visited FIRM node. Thus, the overall feedback $\pi : \mathbb{V} \times \mathbb{B} \to \mathbb{U}$ can be written as:

$$
\pi(B, b) = \pi^g(B)(b) = \mu(b) \tag{9}
$$

Initial controller: Now, let us consider the first step of planning where the system has not visited any FIRM node yet. Given the initial belief is b_0, if b_0 is in a FIRM node B, then we can just follow generated the control signal as $\pi(B, b_0)$ based on Eq. 9. However, if b_0 does not belong to any of the FIRM nodes, we consider a singleton FIRM node $B_0 = \{b_0\}$ and connect it to the graph. Let us denote the set of newly added local controllers by $\mathbb{M}(0)$. Computing the transition cost $C(b_0, \mu^{ij})$, and probabilities $\mathbb{P}(B^j_\gamma|b_0, \mu^{ij})$, and $\mathbb{P}(F|b_0, \mu^{ij})$, for invoking local controllers $\mu^{ij} \in \mathbb{M}(0)$ at b_0, we choose the best initial controller μ^0_* as:

$$
\pi^g(B_0) = \mu^0_* = \arg\min_{\mu^{ij}\in\mathbb{M}(0)} \{C^g(B_0, \mu^{ij}) + \sum_{\gamma=1}^{m} \mathbb{P}^g(B^j_\gamma|B_0, \mu^{ij})J^g(B^j_\gamma) + \mathbb{P}^g(F|B_0, \mu^{ij})J^g(F)\} \tag{10}
$$

Extending π^g to take B_0 into account, we now can use $\pi(B_0, b_0)$ to generate the control signal. It is worth noting that computing μ^0_* is the only part of computation that depends on the initial belief and has to be reproduced for every query with a new initial belief. After computing μ_*^0 we always store the last visited FIRM node and use policy π (computed offline) in Eq. 9 to generate control signals in future time steps.

V. PLQG-BASED FIRM CONSTRUCTION

In this section, we construct a concrete instantiation of the graph described in the previous section. We utilize PLQG controllers to design graph edges and reachable FIRM nodes B^j_γ required in (8). Then we discuss how the transition probabilities $\mathbb{P}^g(\cdot|B^i_\alpha, \mu^{\alpha,ij})$, and costs $C^g(B^i_\alpha, \mu^{\alpha,ij})$ in (8) are computed. In this instantiation we restrict the belief to Gaussian distributions and we start by defining notation needed for dealing with Gaussian beliefs.

Gaussian belief space: Let us denote the estimation vector by x^+, whose distribution is $b_k = p(x^+_k) = p(x_k|z_{0:k})$. Denote the mean and covariance of x^+ by $\widehat{x}^+ = \mathbb{E}[x^+]$ and $P = \mathbb{E}[(x^+ - \widehat{x}^+)(x^+ - \widehat{x}^+)^T]$, respectively. Denoting the Gaussian belief space by $\mathbb{GB}$, every function $b(\cdot) \in \mathbb{GB}$, can be characterized by a mean-covariance pair, i.e., $b \equiv (\widehat{x}^+, P)$. Abusing notation, we also show this using "equality relation", i.e., $b = (\widehat{x}^+, P)$.

A. *Designing PLQG-based Graph Nodes* $\{B^j_\alpha\}$

LQG controllers: A Linear Quadratic Gaussian (LQG) controller is composed of a Kalman filter as the state estimator and a Linear Quadratic Regulator (LQR) as the separated controller [20]. Thus, the belief dynamics $b_{k+1} = \tau(b_k, u_k, z_{k+1})$ come from the Kalman filtering equations, and the controller $u_k = \mu(b_k)$ that acts on the belief, comes from the LQR equations. LQG is an optimal controller for linear systems with Gaussian noise [8]. However, it is most often used for stabilizing nonlinear systems to a given trajectory or to a given point.

Periodic LQG: Periodic LQG (PLQG) is a time-varying LQG that is designed to track a given periodic trajectory [10], [12]. In Appendix I we review the periodic LQG controller in detail. Here, we only state the belief reachability result under the PLQG.

System model and quadratic cost: Consider a T-periodic PNPRM orbit $O = (x^p_k, u^p_k)_{k\geq 1}$ and the set of nodes $\{\mathbf{v}_\alpha\}$ on it. Let us denote the time-varying linear (linearized) system along the orbit O by the tuple $\Upsilon_k = (\mathbf{A}_k, \mathbf{B}_k, \mathbf{G}_k, \mathbf{Q}_k, \mathbf{H}_k, \mathbf{M}_k, \mathbf{R}_k)$ that represents the following state space model, where $\Upsilon_k = \Upsilon_{k+T}$:

$$x_{k+1} = \mathbf{A}_k x_k + \mathbf{B}_k u_k + \mathbf{G}_k w_k, \quad w_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_k) \tag{11a}$$

$$z_k = \mathbf{H}_k x_k + \mathbf{M}_k v_k, \qquad v_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k), \tag{11b}$$

Consider a PLQG controller that is designed for the system in (11) to track the orbit $(x^p_k, u^p_k)_{k\geq 1}$ through minimizing the following quadratic cost:

$$J = \mathbb{E}[\sum_{k\geq 0} \overline{x}_k^T \mathbf{W}_x \overline{x}_k + \overline{u}_k^T \mathbf{W}_u \overline{u}_k] \tag{12}$$

where $\overline{x}_k = x_k - x^p_k$ and $\overline{u}_k = u_k - u^p_k$. Matrices $\mathbf{W}_x$ and $\mathbf{W}_u$ are positive definite weight matrices for state and control cost, respectively. Let us also define matrices $\check{\mathbf{Q}}_k$ and $\check{\mathbf{W}}_x$ such that $\mathbf{G}_k\mathbf{Q}_k\mathbf{G}_k^T = \check{\mathbf{Q}}_k\check{\mathbf{Q}}_k^T$, $\mathbf{W}_x = \check{\mathbf{W}}_x^T\check{\mathbf{W}}_x$, for all k. Now, consider the class of systems, and associated PLQG controllers that satisfy the following property.

Property 1: The pairs $(\mathbf{A}_k, \mathbf{B}_k)$ and $(\mathbf{A}_k, \check{\mathbf{Q}}_k)$ are controllable pairs [8], and the pairs $(\mathbf{A}_k, \mathbf{H}_k)$ and $(\mathbf{A}_k, \check{\mathbf{W}}_x)$ are observable pairs [8], for all $k = 1, \cdots, T$.

Belief node reachability under PLQG: In the following, we present three lemmas, through which we can construct pairs of periodic LQG controllers, and reachable nodes in belief space, for non-stoppable/nonholonomic dynamical systems.

Lemma 1: **(Cyclostationary behavior of belief under PLQG)** Consider the PLQG controller designed for the system in (11) to track the orbit $(x^p_k, u^p_k)_{k\geq 1}$. Given Property 1 is satisfied, in the absence of stopping region, the belief process b_k under PLQG converges to a Gaussian cyclostationary process [9], i.e., the distribution over belief converges to a T-periodic Gaussian distribution, where we denote the mean and covariance of this process by b^c_k and $\mathtt{C}_k$, respectively:

$$b_k \sim \mathcal{N}(b^c_k, \mathtt{C}_k) = \mathcal{N}(b^c_{k+T}, \mathtt{C}_{k+T}), \tag{13}$$

where $b_k \equiv (\widehat{x}^+_k, P_k)$ and $b^c_k \equiv (x^p_k, \check{P}_k)$. The covariance matrices $\check{P}_k$ is characterized in Lemma 2 and covariance $\mathtt{C}_k$ is characterized in Appendix I (Eq. 68).

Proof: See Appendix I. ■

Lemma 2: **(Convergence of DPRE)** Given Property 1, the following Discrete Periodic Riccati Equation (DPRE) has a unique Symmetric T-Periodic Positive Semi-definite (SPPS) solution [10], denoted by $\check{P}^-_k$:

$$\check{P}^-_{k+1} = \mathbf{A}_k(\check{P}^-_k - \check{P}^-_k\mathbf{H}_k^T(\mathbf{H}_k\check{P}^-_k\mathbf{H}_k^T + \mathbf{M}_k\mathbf{R}_k\mathbf{M}_k^T)^{-1}\mathbf{H}_k\check{P}^-_k)\mathbf{A}_k^T + \mathbf{G}_k\mathbf{Q}_k\mathbf{G}_k^T \tag{14}$$

Moreover, the covariance matrix $\check{P}_k$ introduced in Lemma 1 is computed as

$$\check{P}_k = \check{P}^-_k - \check{P}^-_k\mathbf{H}_k^T(\mathbf{H}_k\check{P}^-_k\mathbf{H}_k^T + \mathbf{M}_k\mathbf{R}_k\mathbf{M}_k^T)^{-1}\mathbf{H}_k\check{P}^-_k \tag{15}$$

Proof: See [10]. ■

Now, we state the main result, through which we can construct the proper pairs of periodic LQG controller and reachable nodes in belief space.

Lemma 3: **(Belief node reachability under PLQG)** Consider the PLQG controller μ designed for the system in (11) to track the orbit $(x^p_k, u^p_k)_{k\geq 1}$. Suppose the matrix $\mathbf{H}_k$ is full rank, and Property 1 is satisfied. Also, consider the sets $B_1, B_2, \cdots, B_m$ in belief space, such that interior of B_α contains $b^c_{k_\alpha}$ for some $k_\alpha \in \{1, \cdots, T\}$. Then, under μ, the region $\cup_\alpha B_\alpha$ is reachable in finite time with probability one.

Proof: The intuitive idea behind the proof is: if we define a region centered at the mean value of a Gaussian distribution, and if we sample from this distribution, in a finite number of samples we will end up with a sample in the given region. The rigorous proof is detailed in Appendix II. ■

FIRM nodes: As mentioned, to construct a graph in belief space we first construct its underlying PNPRM, characterized by the triple $\{\{O^j\}, \{\mathbf{v}^j_\alpha\}, \{\mathbf{e}_{ij}\}\}$. Linearizing the system along the j-th orbit $O^j = (x^{p^j}_k, u^{p^j}_k)_{k\geq 0}$ results in a time-varying T-periodic system $\Upsilon^j_k = (\mathbf{A}^j_k, \mathbf{B}^j_k, \mathbf{G}^j_k, \mathbf{Q}^j_k, \mathbf{H}^j_k, \mathbf{M}^j_k, \mathbf{R}^j_k)$:

$$x_{k+1}= \mathbf{A}^j_k x_k + \mathbf{B}^j_k u_k + \mathbf{G}^j_k w_k, \quad w_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}^j_k) \tag{16a}$$

$$z_k= \mathbf{H}^j_k x_k + \mathbf{M}^j_k v_k, \qquad\qquad v_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}^j_k). \tag{16b}$$

where w_k and v_k are motion and measurement noises, respectively, drawn from zero-mean Gaussian distributions with covariances $\mathbf{Q}^j_k$ and $\mathbf{R}^j_k$. Since the system in (16) is T-periodic (i.e.,$\Upsilon^j_k = \Upsilon^j_{k+T}$), we can design a corresponding PLQG controller μ^j_k. The controller μ^j_k is referred to as the j-th *node-controller*. Since the orbits are designed such that Property 1 is satisfied on them, based on Lemma 1 the belief converges to a Gaussian cyclostationary process. The mean of this cyclostationary process is denoted by $b^{c^j}_k$ and is characterized in Lemma 2, where its existence and uniqueness are guaranteed.

Corresponding to the PNPRM node $\mathbf{v}^j_\alpha$ on orbit O^j we choose the belief nodes B^j_α as an ϵ-neighborhood of $\grave{b}^j_\alpha := b^{c^j}_{k_\alpha} \equiv (\mathbf{v}^j_\alpha, \check{P}^j_{k_\alpha})$: (See Fig.2.)

$$B^j_\alpha = \{b \equiv (x, P) : \|x - \mathbf{v}^j_\alpha\| < \delta_1, \|P - \check{P}^j_{k_\alpha}\|_m < \delta_2\}, \tag{17}$$

where $\|\cdot\|$ and $\|\cdot\|_m$ denote suitable vector and matrix norms, respectively. The size of FIRM nodes are determined by δ_1 and δ_2. Based on Lemma 3, $\cup_\alpha B^j_\alpha$ is a reachable region under node-controller μ^j_k. Note that δ_1 and δ_2 need to be sufficiently small to satisfy the approximation in (4).

B. PLQG-based Graph Edges $\{\mu^{\alpha,ij}\}$

The role of the local controller $\mu^{\alpha,ij}$ is to drive the belief from the node B^i_α to $\cup_\gamma B^j_\gamma$, i.e., to a node B^j_γ on the j-th orbit. To construct the local controller $\mu^{\alpha,ij}$, we precede the node-controller μ^j_k, with a time-varying LQG controller $\overline{\mu}^{\alpha,ij}_k$, which is called the *edge-controller* here.

Edge-controller: Consider a finite trajectory that consists of three segments: *i)* the pre-edge $\mathbf{e}^{i_\alpha j}$ as defined in Section II, *ii)* the edge itself $\mathbf{e}^{ij}$, and *iii)* a part of O^j that connects the ending point of $\mathbf{e}^{ij}$ to $x^{p^j}_0$. Edge-controller $\overline{\mu}^{i_\alpha j}_k$ is a time-varying LQG controller that is designed to track this finite trajectory. The main role of the edge-controller is that it takes the belief at node B_i and drives it to the vicinity of a starting point of orbit O^j, where it hands over the system to the node-controller, and node-controller in turn takes the system to a FIRM node.

Local controllers: Thus, overall, the local controller (or graph edge in belief space) $\mu^{\alpha,ij}$ is the concatenation of the edge-controller $\overline{\mu}^{\alpha,ij}_k$ and the node-controller μ^j_k. Note that since reachability is guaranteed by the node-controller (PLQG), by this construction, the stopping region $\cup_\gamma B^j_\gamma$ is also reachable under the local controller $\mu^{\alpha,ij}$.

C. Transition Probabilities and Costs

In general, it can be a computationally expensive task to compute the transition probabilities $\mathbb{P}(\cdot|B^i_\alpha, \mu^{\alpha,ij})$ and costs $C(B^i_\alpha, \mu^{\alpha,ij})$ associated with invoking local controller $\mu^{\alpha,ij}$ at node B^i_α. However, owing to the offline construction of FIRM, it is not an issue in FIRM. We utilize sequential Monte-Carlo methods [14] to compute the collision and absorption probabilities. In other words, for each graph edge we simulate the execution of the corresponding local controller for M times and accordingly approximate the probability of reaching the nodes on the target orbit as well as probability of hitting the failure set along the way. This process is done offline.

Depending on the application, a suitable transition cost can be defined. In this paper, we consider a measure of estimation accuracy as the transition cost along the edges. This leads to a planner that favors paths, on which the estimator and consequently the controller can perform better. A measure of estimation error we use here is the trace of estimation covariance; i.e., $\Phi^{\alpha,ij} = \mathbb{E}[\sum_{k=1}^{\mathcal{T}} \text{tr}(P^{\alpha,ij}_k)]$, where $P^{\alpha,ij}_k$ is the estimation covariance at the k-th time step of the execution of local controller $\mu^{\alpha,ij}$. The outer expectation operator is useful in dealing with the Extended Kalman Filter (EKF), whose covariance is stochastic [13], [24]. Moreover, as we are also interested in faster paths, we take into account the corresponding mean stopping time, i.e., $\widehat{\mathcal{T}}^{\alpha,ij} = \mathbb{E}[\mathcal{T}^{\alpha,ij}]$, and the total cost of invoking $\mu^{\alpha,ij}$ at B^i_α is considered as a linear combination of estimation accuracy and expected stopping time, with suitable scalar coefficients ξ_1 and ξ_2.

$$C(B^i_\alpha, \mu^{\alpha,ij}) = \xi_1 \Phi^{\alpha,ij} + \xi_2 \widehat{\mathcal{T}}^{\alpha,ij}. \tag{18}$$

D. Construction of PLQG-FIRM and Planning With it

Offline construction of FIRM: The crucial feature of FIRM is that it can be constructed offline and stored, independent of future queries. Note that based on Algorithms 1 and 2, we still need to know the goal location. However, to be fully independent of both start and goal location of the query, one can solve the DP in the online phase. Moreover, owing to the reduction from the original POMDP to a dynamic programming on the graph nodes, we can solve DP in 8 using standard DP techniques such as value/policy iteration to get the optimal graph policy π^g. Algorithm 1 details the offline construction of FIRM graph.

Online planning with FIRM: Since the FIRM graph is computed offline, the online phase of planning (and replanning) on the roadmap becomes very efficient. If the given initial belief b_0 does not belong to any B_i, we create a singleton set $B_0 = b_0$. To connect the B_0 to FIRM, we first, compute the expected value of the robot state, i.e. $\mathbb{E}[x_0]$ using its distribution b_0 and add the $\mathbb{E}[x_0]$ to the underlying PNPRM nodes. The set of newly added edges going from $\mathbb{E}[x_0]$ to the nodes on PNPRM are denoted as $\mathcal{E}(0)$. We design the local controllers associated with each edge in $\mathcal{E}(0)$ and call the set of them as $\mathbb{M}(0)$. Then choosing a local controller in $\mathbb{M}(0)$, the belief enters one of FIRM nodes, if no collision occurs. Thus, given the current node, we use policy π^g defined in (8b) over FIRM nodes to find μ^*, and pick μ^* to move the robot into $B(\mu^*)$. Algorithm 2 illustrates this procedure.

Computational complexity of offline graph construction: Consider an underlying PNPRM with N orbits, m nodes on each orbit, and degree k; i.e., each orbit in PNPRM is connected to k nearest neighboring orbits. Thus, overall it has mN nodes and Nk orbit edges. In the offline phase we need to leverage PNPRM orbits and edges to FIRM orbits and edges in belief space. *(i)* Extension of PNPRM orbits to belief space consists of a constant computation of solving two Riccati equations and designing corresponding PLQG controller. Denoting the computation complexity of this process by c_n, the computational complexity of extending PNPRM orbits to FIRM orbits is of the order $O(c_n N)$. *(ii)* Extension of each PNPRM edge to belief space consists of evaluating the performance of its corresponding local controller and computing transition probabilities and costs. Let us denote the cost of this process by c_e. In a PNPRM with degree k, we have a Nk edges and corresponding to each PNPRM edge, we have m FIRM edges. Thus, the computational complexity of extending edges to belief space is $O(c_e mNk)$. So, overall the offline computational complexity is $O(c_n N + c_e mNk)$. The complexity of each iteration in value iteration algorithm is $O(|\mathbb{V}|^2|\mathbb{M}|)$, where $|\mathbb{V}| = mN$ nodes and $|\mathbb{M}| = mNk$. However, in practice the dominating factor is the extension of edges to belief space because the constant multiplier c_e in general is large. If the Monte Carlo simulation is chosen to evaluate the edge costs and transition probabilities, c_e will increase linearly in the number of particles utilized in the Monte Carlo simulation as well as the number of constraints and how they are being evaluated.

Computational complexity of online planning with graph: As discussed in Section IV, the only part that needs to be done online is the computation of first local controller (See Eq. 10). To do so, we need to evaluate k edges only. Thus, the computational complexity of online planning with FIRM is $O(c_e k)$. This computation occurs once in the beginning of planning. The rest of planning is just plugging last visited FIRM node B and current belief b into the planner π (See Eq. 9) and generating the control signal u_k.

VI. Experimental Results

In this section we present simulation results for two different types of robots: a planar robot whose motion is described by a unicycle model and a 6 DoF small aerial vehicle subject to rigid body kinematics. The robots are equipped with exteroceptive sensors that provide range and bearing measurements from existing radio beacons (landmarks) in the environment.

A. 2D Unicycle Model

Here, we illustrate the results of FIRM construction on a simple PNPRM.

Motion model: As a motion model, we consider the nonholonomic unicycle model which has the following kinematics:

$$x_{k+1} = f(x_k, u_k, w_k) = \begin{pmatrix} \mathsf{x}_k + (V_k + n_v)\delta t \cos\theta_k \\ \mathsf{y}_k + (V_k + n_v)\delta t \sin\theta_k \\ \theta_k + (\omega_k + n_\omega)\delta t \end{pmatrix}, \tag{19}$$

where $x_k = (\mathsf{x}_k, \mathsf{y}_k, \theta_k)^T$ describes the robot state (2D position and heading angle). The vector $u_k = (V_k, \omega_k)^T$ is the control vector consisting of linear velocity V_k and angular velocity ω_k. The motion noise vector is denoted by $w_k = (n_v, n_\omega)^T \sim \mathcal{N}(0, \mathbf{Q}_k)$.

Observation model: The i-th landmark is denoted by L^i. Denoting the vector from robot to the i-th landmark by ${}^i\mathbf{d} = [{}^id_x, {}^id_y]^T := L^i - \mathsf{p}$, where $\mathsf{p} = [\mathsf{x}, \mathsf{y}]^T$ is the position of the robot. Measuring L^i can be modeled as follows:

$${}^iz = {}^ih(x, {}^iv) = [\|{}^i\mathbf{d}\|, \text{atan2}({}^id_y, {}^id_x) - \theta]^T + {}^iv, \tag{20}$$

where, atan2$(\cdot,\cdot)$ is the four-quadrant inverse tangent function. Observation noise is drawn from a zero-mean Gaussian distribution ${}^iv \sim \mathcal{N}(\mathbf{0}, {}^i\mathbf{R})$ where ${}^i\mathbf{R} = \text{diag}((\eta_r\|{}^i\mathbf{d}\| + \sigma_b^r)^2, (\eta_\theta\|{}^i\mathbf{d}\| + \sigma_b^\theta)^2)$. Function "diag" returns a square block-diagonal matrix by placing its inputs on the main diagonal. The uncertainty (standard deviation) of sensor reading increases

Algorithm 1: Offline Construction of PLQG-FIRM

input : State space $\mathbb{X}$, constraints set F, belief space $\mathbb{B}$

output : FIRM graph G

Construct a PNPRM with T-periodic orbits $\mathcal{O} = \{O^j = (x_k^{p^j}, u_k^{p^j})_{k\geq 0}\}$, nodes $\mathcal{V} = \{\mathbf{v}_\alpha^j\}$, and edges $\mathcal{E} = \{\mathbf{e}^{ij}\}$, where $i, j = 1, \cdots, n$ and $\alpha = 1, \cdots, m$;

foreach *PNPRM orbit* $O^j \in \mathcal{O}$ **do**

- Design the node-controller (periodic LQG) μ_k^j along the periodic trajectory;
- Compute the periodic mean belief trajectory $b_k^{c^j} \equiv (x_k^{p^j}, \check{P}_k^j)$ using (15);
- Construct m FIRM nodes $\mathbb{V}^j = \{B_1^j, \cdots, B_m^j\}$ using (17), where B_α^j is centered at $b_{k_\alpha}^{c^j}$;

Collect all FIRM nodes $\mathbb{V} = \cup_{j=1}^n \mathbb{V}^j$;

foreach $(B_\alpha^i, \mathbf{e}^{ij})$ *pair* **do**

- Design the edge-controller $\overline{\mu}_k^{\alpha,ij}$, as discussed in Section V-B;
- Construct the local controller $\mu_k^{\alpha,ij}$ by concatenating edge-controller $\overline{\mu}_k^{\alpha,ij}$ and node-controller μ_k^j;
- Set the initial belief b_0 equal to the center of B_α^i, based on the approximation in (4);
- Generate (in simulation) sample belief paths $b_{0:\mathcal{T}}$ and state paths $x_{0:\mathcal{T}}$ induced by controller $\mu^{\alpha,ij}$ invoked at B_α^i;
- Compute the transition probabilities $\mathbb{P}^g(F|B_\alpha^i, \mu^{\alpha,ij})$ and $\mathbb{P}^g(B_\gamma^j|B_\alpha^i, \mu^{\alpha,ij})$ for all γ and transition cost $C^g(B_\alpha^i, \mu^{\alpha,ij})$ based on the simulated trajectories (see Section V-C);

Collect all local controllers $\mathbb{M} = \{\mu^{\alpha,ij}\}$;

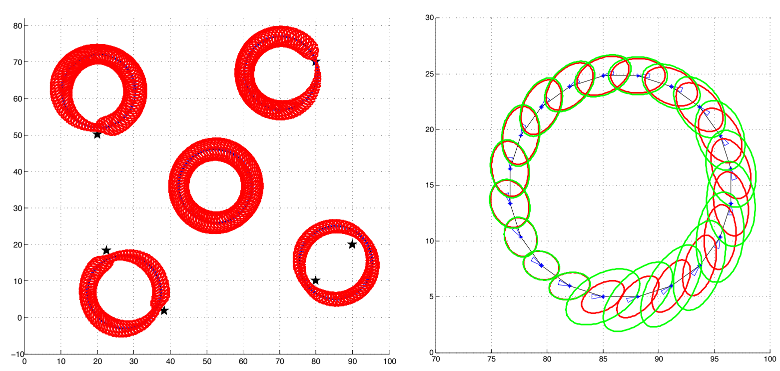
Compute cost-to-go J^g and feedback π^g over the FIRM by solving the DP in (8);

$G = (\mathbb{V}, \mathbb{M}, J^g, \pi^g)$;

return G;

as the robot gets farther from the landmarks. The parameters $\eta_r = \eta_\theta = 0.3$ determine this dependency, and $\sigma_b^r = 0.01$ meter and $\sigma_b^\theta = 0.5$ degrees are the bias standard deviations. A similar model for range sensing is used in [23]. The robot observes all N_L landmarks at all times and their observation noises are independent. Thus, the total measurement vector is denoted by $z = [{}^1z^T, {}^2z^T, \cdots, {}^{N_L}z^T]^T$ and due to the independence of measurements of different landmarks, the observation model for all landmarks can be written as $z = h(x) + v$, where $v \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$ and $\mathbf{R} = \mathrm{diag}({}^1\mathbf{R}, \cdots, {}^{N_L}\mathbf{R})$.

We first show a typical SPPS solution of DPRE on the orbits. Fig. 3(a) shows a simple environment with six radio beacons (black stars). For illustration purposes, we choose five large circular orbits and every orbit is discretized to 100 steps. Thus the SPPS solution of the DPRE in (14) on each orbit leads to hundred covariance matrices that are superimposed on the graph in red. As is seen from Fig. 3(a), the localization uncertainty along the orbit is not homogeneous and varies periodically. Another important observation from the Fig. 3(a) is obtained by noticing the left top orbit in the Fig. 3(a). As it can be seen, the localization uncertainty (covariance ellipse) in the left and right hand sides of the landmark are not symmetric (the right hand side is larger than the left hand side). In other words, two points on an orbit with the same distance from landmarks (i.e., with the same observation noise) might have different localization uncertainty, which emphasizes the role of the dynamics model in filtering and its interaction with the observation model. In Fig. 3(b), we illustrate the covariance



(a) (b)

Fig. 3. (a) Five orbits ($T = 100$) and corresponding periodic estimation covariances as the SPPS solution of DPRE in (14). (b) Sample covariance convergence on an orbit ($T = 20$) under PLQG. Red ellipses are the solution of DPRE and green ellipses are the evolution of estimation covariance. The initial covariance is three times bigger than the SPPS solution of DPRE, i.e., $P_0 = 3\check{P}_0$.

Algorithm 2: Online Phase Algorithm (Planning with PLQG-based FIRM)

input : Initial belief b_0, FIRM graph G, Underlying PNPRM graph
if $\exists B^i_\alpha \in \mathbb{V}$ *such that* $b_0 \in B^i_\alpha$ **then**
  Choose the next local controller $\mu^{\alpha,ij} = \pi^g(B^i_\alpha)$;
else
  Compute $\mathbf{v}_0 = \mathbb{E}[x_0]$ based on b_0, and connect $\mathbf{v}_0$ to the PNPRM orbits. Call the set of newly added edges $\mathcal{E}(0) = \{\mathbf{e}^{0j}\}$;
  Design local planners associated with edges in $\mathcal{E}(0)$; Collect them in set $\mathbb{M}(0) = \{\mu^{0,0j}\}$;
  foreach $\mu \in \mathbb{M}(0)$ **do**
    Generate sample belief and state paths $b_{0:\mathcal{T}}$, $x_{0:\mathcal{T}}$ induced by taking μ at b_0;
    Compute the transition probabilities $\mathbb{P}(\cdot|b_0,\mu)$ and transition costs $C(b_0,\mu)$;
  Set $\alpha, i = 0$; Choose the best initial local planner $\mu^{0,0j}$ within the set $\mathbb{M}(0)$ using (10);
while $B^i_\alpha \neq B_{goal}$ **do**
  while ($\nexists B^j_\gamma, s.t.,\ b_k \in B^j_\gamma$) *and "no collision"* **do**
    Apply the control $u_k = \mu^{\alpha,ij}_k(b_k)$ to the system;
    Get the measurement z_{k+1} from sensors;
    if *Collision happens* **then return** Collision;
    Update belief as $b_{k+1} = \tau(b_k, \mu^{\alpha,ij}_k(b_k), z_{k+1})$;
  Update the current FIRM node $B^i_\alpha = B^j_\gamma$;
  Choose the next local controller $\mu^{\alpha,ij} = \pi^g(B^i_\alpha)$;

convergence in the periodic belief process. As can be seen in Fig. 3(b), the initial covariance is three times larger than the limiting covariance, and in less than one period it converges to the SPPS solution of DPRE. The convergence time is a random quantity, whose mean and variance can be estimated through simulations. However, in practical cases it usually converges in less than one full period, because the initial covariance is closer to the actual solution (due to the use of edge-controllers) and also the orbit size is much smaller, when compared to Fig. 3(b).

Figure 4(a) shows a sample PNPRM with 23 orbits and 67 edges. To simplify the explanation of the results, we assume $m = 1$, i.e., we choose one node on each orbit. All elements in Fig.4(a) are defined in $(\mathtt{x}, \mathtt{y}, \theta)$ space but only the $(\mathtt{x}, \mathtt{y})$ portion is shown here. To construct the FIRM nodes, we first solve the corresponding DPREs on each orbit and design its corresponding node-controller (PLQG). Then, we pick the node centers $\grave{b}^j_\alpha = (\mathbf{v}^j_\alpha, \check{P}^j_{k_\alpha})$ and construct the FIRM nodes based on the component-wise version of (17), to handle the error scale difference in position and orientation variables:

$$B^j_\alpha = \{b = (x, P) |\ \ |x - \mathbf{v}^j_\alpha| \dot{<} \epsilon, |P - \check{P}^j_{k_\alpha}| \dot{<} \Delta\}, \tag{21}$$

where $|\cdot|$ and $\dot{<}$ stand for the absolute value and component-wise comparison operators, respectively. We set $\epsilon = [0.8, 0.8, 5°]^T$ and $\Delta = \epsilon\epsilon^T$ to quantify the B^j_α's.

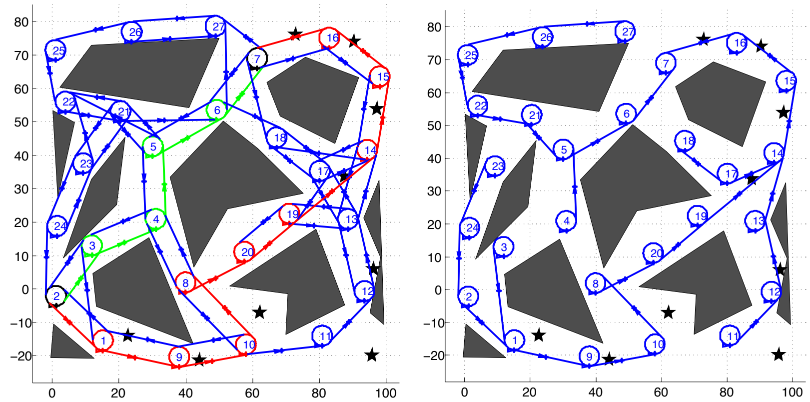
After designing FIRM nodes and local controllers, the transition costs and probabilities are computed in the offline construction phase. Here, we use sequential weighted Monte-Carlo based algorithms [14] to compute these quantities. In other words, for every $(B^i_\alpha, \mu^{\alpha,ij})$ pair, we perform M runs and accordingly approximate the transition probabilities $\mathbb{P}^g(B^j_\gamma|B^i_\alpha, \mu^{\alpha,ij})$, $\mathbb{P}^g(F|B^i_\alpha, \mu^{\alpha,ij})$, and costs $C^g(B^i_\alpha, \mu^{\alpha,ij})$. A similar approach is detailed in [4]. Table I shows these quantities for several $(B^i_\alpha, \mu^{\alpha,ij})$ pairs corresponding to Fig.4(a), where $M = 101$ and the coefficients in (18) are $\xi_1 = 0.98$ and $\xi_2 = 0.02$.

TABLE I

COMPUTED COSTS FOR SEVERAL PAIRS OF NODE-AND-CONTROLLER USING 101 PARTICLES.

$(B^i_\alpha,\mu^{\alpha,ij})$ pair	$B^2_1,\mu^{1,(2,3)}$	$B^4_1,\mu^{1,(4,5)}$	$B^6_1,\mu^{1,(6,7)}$	$B^{11}_1,\mu^{1,(11,12)}$	$B^2_1,\mu^{1,(2,1)}$	$B^8_1,\mu^{1,(8,20)}$	$B^{16}_1,\mu^{1,(16,7)}$
$\mathbb{P}^g(F\|B^i_\alpha,\mu^{\alpha,ij})$	9.9010%	17.8218%	15.8416%	29.7030%	7.9208%	1.9802%	0.9901%
$\Phi^{\alpha,ij}$	2.1386	2.2834	1.9181	0.9152	2.1695	1.1857	0.4385
$\mathbb{E}[\mathcal{T}^{\alpha,ij}]$	63.6703	82.6747	62.5882	58.2000	51.7033	50.2755	35.4653

Plugging the computed transition costs and probabilities into (8), we can solve the DP problem and compute the policy π^g on the graph. This process is performed only once offline, independent of the starting point of the query. Fig. 4(b) shows the policy π^g on the constructed FIRM in this example. At every FIRM node B^i_α, the policy π^g decides which local controller needs to be invoked, which in turn aims to take the robot belief to the next FIRM node. It is worth noting that if we had more than one node on each orbit, the feedback π^g may return different controllers for each of them and for every orbit we may have more than one outgoing arrow in Fig. 4(b).



(a) (b)

Fig. 4. A sample PNPRM with circular orbits. Number of each orbit is written at its center. Nine landmarks (black stars) and obstacles (gray polygons) are also shown. The directions of motion on orbits and edges are shown by little triangles with a cross in their heading direction. (a) Orbits 2 and 7 (distinguished in black) are start and goal nodes, respectively. Shortest path (green) and the most-likely path (red) under FIRM policy are also shown. (b) Assuming on each orbit, there exists a single node, the feedback π^g is visualized for all FIRM nodes.

As discussed, the online part of planning is very efficient as it only requires executing the controller and generating the control signal. Moreover, if due to some unmodeled large disturbances, the system deviates significantly from the planned path, it suffices to bring the system back to the closest FIRM node and from thereon the optimal plan is already known, i.e., π^g drives the robot to the goal region as shown in Fig. 4(b).

We show the most likely path under the π^g in red in Fig. 4(a). The shortest path is also illustrated in Fig. 4(a) in green. It can be seen that the "most likely path under the best policy" detours from the shortest path to a path along which the filtering uncertainty is smaller, and it is easier for the controller to avoid collisions.

B. 6 DoF Aircraft Model

In this section, we consider a surveillance application for a small fixed wing aerial vehicle. We assume that targets to monitor are submitted from the control station frequently. Each time a new target is submitted, the aircraft has to replan in real-time and go toward the new goal, while minimizing the collision probability and the costs associated with the task objective.

System state: The system considered in this experiment is a robot with 6 Degrees of Freedom (DoF). The motion is the rigid body 6 DoF kinematics. The state of the robot x_k at time k is composed of its 3D position in Cartesian coordinates $\mathbf{p}_k$ described in the ground (inertial) frame and its orientation $\mathbf{q}_k$, which is encoded by quaternions.

$$x_k = [\mathbf{p}_k^T, \mathbf{q}_k^T]^T = [\mathrm{x_k}, \mathrm{y_k}, \mathrm{z_k}, q_{0_k}, q_{1_k}, q_{2_k}, q_{3_k}]^T, \tag{22}$$

where

$$\mathbf{p}_k = [\mathrm{x_k}, \mathrm{y_k}, \mathrm{z_k}]^T, \mathbf{q}_k = [q_{0_k}, q_{1_k}, q_{2_k}, q_{3_k}]^T, \tag{23}$$

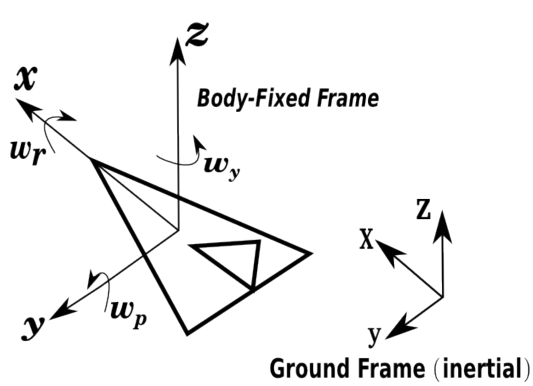


Fig. 5. The aircraft attached body fixed frame and ground frame

Remark: Let us denote the control error (the difference between the desired state x_k^p and the mean of estimated state $\widehat{x}_k^+$) by $\widehat{e}_k^+$. In computing the control error, one can directly subtract the positional part of the state vector. However, the error in orientation (quaternions) $\mathbf{q}$ and $\mathbf{q}'$ is calculated as $\delta\mathbf{q} = \mathbf{q} \otimes \text{inv}(\mathbf{q}')$ where $\otimes$ and $\text{inv}(\cdot)$ denote the quaternion multiplication and inversion operators, respectively. We set $\delta q_0 = 0$ in calculating the control error. This is valid for small rotations since a change in the scalar part of the quaternion does not provide information about the direction of the rotation vector. Further, since we know that for a quaternion $\mathbf{q}$, $q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1$, by controlling q_1, q_2, q_3 we implicitly control q_0.

Motion Model: Let f be the state transition function such that,

$$x_{k+1} = f(x_k, u_k, w_k), \tag{24}$$

where, the control vector u_k is composed of the vehicle's linear velocity V_k along body x-axis and angular velocities about the body axes.

$$u_k = [V_k, \omega_k^r, \omega_k^p, \omega_k^y], \tag{25}$$

in which, ω^r, ω^p, and ω^y are the roll, pitch, and yaw rates, respectively. The motion noise is denoted by $w_k = (n_v, n_{\omega^r}, n_{\omega^p}, n_{\omega^y})^T \sim \mathcal{N}(0, \mathbf{Q}_k)$. In our simulations, $\mathbf{Q} = \text{diag}((\eta_V V + \sigma_b^V)^2, (\eta_{\omega^r}\omega^r + \sigma_b^{\omega^r})^2, (\eta_{\omega^p}\omega^p + \sigma_b^{\omega^p})^2), (\eta_{\omega^y}\omega^y + \sigma_b^{\omega^y})^2)$, where the parameters are $\eta_V = \eta_{\omega^r} = \eta_{\omega^p} = \eta_{\omega^y} = 0.005$, $\sigma_b^V = 0.02$ meters, $\sigma_b^{\omega^r} = \sigma_b^{\omega^p} = \sigma_b^{\omega^y} = 0.25$ degrees. To describe the kinematics model, we split the motion model into two parts: position and orientation (attitude).

To derive a model that governs the position of the robot (i.e., $\mathbf{p}_{k+1} = f_p(\mathbf{p}_k, \mathbf{q}_k, u_k, w_k)$), we first need to transform velocity V_k from body to the ground frame. We denote the velocity in the body fixed frame as ${}^bV$ and in the inertial (ground) frame as ${}^gV$. Thus,

$${}^gV = R_{gb}{}^bV, \tag{26}$$

where, ${}^bV = [V, 0, 0]^T$ and R_{gb} is the rotation matrix that transforms the body frame to the ground frame. In terms of the quaternions, the R_{gb} matrix is as follows:

$$R_{gb}(\mathbf{q}) = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix}. \tag{27}$$

Similarly, we transform the motion noise in velocity to the ground frame,

$${}^gn_V = R_{gb}{}^bn_V, \tag{28}$$

where ${}^bn_V = [n_V, 0, 0]^T$. Therefore, f_p can be described as:

$$\mathbf{p}_{k+1} = f_p(\mathbf{p}_k, \mathbf{q}_k, u_k, w_k) = \mathbf{p}_k + {}^gV\delta t + {}^gn_V\sqrt{\delta t} = \mathbf{p}_k + R_{gb}(\mathbf{q})({}^bV\delta t + {}^bn_V\sqrt{\delta t}). \tag{29}$$

Now, we discuss the model we utilize to govern the orientation of the robot (i.e., $\mathbf{q}_{k+1} = f_q(\mathbf{q}_k, u_k, w_k)$). We start by the quaternion-based attitude kinematics in its continuous-time form that can be written as $\dot{\mathbf{q}} = A\omega$, where $\omega = [\omega^r, \omega^p, \omega^y]^T$ is the angular velocity vector of the robot with respect to the inertial frame expressed in the body frame, and A is given by:

$$A = \frac{1}{2}\begin{bmatrix} -q_1 & -q_2 & -q_3 \\ q_0 & -q_3 & q_2 \\ q_3 & q_0 & -q_1 \\ -q_2 & q_1 & q_0 \end{bmatrix}. \tag{30}$$

Therefore, the discrete version of the quaternion evolution (before sign check) is as follows:

$$\mathbf{q}_{k+1}^s = f_q^s(\mathbf{q}_k, u_k, w_k) = \mathbf{q}_k + \dot{\mathbf{q}}\delta t + n_q, \tag{31}$$

where,

$$n_q = A[n_{\omega^r}, n_{\omega^p}, n_{\omega^y}]^T\sqrt{\delta t}. \tag{32}$$

However, to avoid discontinuity in the control error $\hat{e}_k^+$, we keep the scalar part of quaternion positive; i.e. the next quaternion is:

$$\mathbf{q}_{k+1} = \bar{f}(\mathbf{q}_{k+1}^s) = \mathbf{q}_{k+1}^s\text{sign}(\mathbf{q}_{0_{k+1}}^s), \tag{33}$$

where $\text{sign}(\mathbf{q}_{0_{k+1}}^s)$ is 1 if $\mathbf{q}_{0_{k+1}}^s \geq 0$, and is -1 otherwise. This procedure leads to the smaller angle since $q_0 = cos(\phi/2)$ where ϕ is the magnitude of rotation, and thus, the smaller angular difference (i.e., $|\phi| < \pi$) always leads to a positive q_0. Note that we are allowed to do this because quaternions are invariant to sign; i.e., $\mathbf{q}_{k+1}$ and $-\mathbf{q}_{k+1}$ represent the same orientation. Thus overall we get $\mathbf{q}_{k+1} = f_q(\mathbf{q}_k, u_k, w_k) = \bar{f}(f_q^s(\mathbf{q}_k, u_k, w_k))$.

Finally, since the quaternions norm is constrained (i.e., $\|\mathbf{q}\| = 1$), if the result of an approximate calculation such as *linearized* Kalman filter is a quaternion q that does not satisfy this constraint, we apply the transformation $\mathbf{q}_{k+1} =$

$\|\mathbf{q}_{k+1}\|^{-1}\mathbf{q}_{k+1} = g(\mathbf{q})$. Note that the g is applied on mean and its first order approximation is applied on covariance of quaternion estimation.

Observation Model: The 3D location of the i-th Landmark is defined as $L^i = [L^i_x, L^i_y, L^i_z]$. Denoting the relative vector from robot to landmark L^i by ${}^i\mathbf{d}^g = [{}^id^g_x, {}^id^g_y, {}^id^g_z]^T := L^i - \mathbf{p}$, where $\mathbf{p} = [\mathsf{x}, \mathsf{y}, \mathsf{z}]^T$ is the position of the robot in the ground frame. The relative vector ${}^i\mathbf{d}^g$ needs to be rotated from the ground frame to the body frame by the rotation matrix $R_{bg} = R^T_{gb}$. Thus, ${}^i\mathbf{d}^b = R_{bg}{}^i\mathbf{d}^g$.

The measurement L^i can be modeled as follows:

$$
{}^iz = h(x, {}^iv) = [{}^ir, {}^i\alpha, {}^i\beta]^T = [\|{}^i\mathbf{d}^b\|, \mathrm{atan2}({}^id^b_y, {}^id^b_x), \mathrm{atan2}({}^id^b_z, {}^id^b_x)]^T + {}^iv, \tag{34}
$$

where, ${}^iv \sim \mathcal{N}(0, \mathbf{R}^i)$ and $\mathbf{R}^i = \mathrm{diag}((\eta_r\|{}^i\mathbf{d}\| + \sigma^r_b)^2, (\eta_\alpha\|{}^i\mathbf{d}\| + \sigma^\alpha_b)^2, (\eta_\beta\|{}^i\mathbf{d}\| + \sigma^\beta_b)^2)$. The parameters are $\eta_r = 0.01$, $\eta_\alpha = \eta_\beta = 0.3$ and $\sigma^r_b = 0.01$ meter and $\sigma^\alpha_b = \sigma^\beta_b = 0.5$ degrees are the bias standard deviations.

PNPRM generation: To generate the underlying PNPRM, we need to sample orbits and connect them to each other. In this experiment, we consider circular (counter-clockwise) orbits that are parallel to the ground. To sample an orbit, we sample a random point p^c in 3D space as the orbit center, and generate a circular trajectory with a given maximum yaw rate centered at p^c. More details on this construction can be found in [1]. Finally, we choose three nodes on each orbit uniformly distributed along the orbit.

The edge connecting node $\mathbf{v}^i_\alpha$ to orbit O^j is composed of two segments: pre-edge $\mathbf{e}^{i\alpha j}$ and orbit-edge $\mathbf{e}^{ij}$. The edge $\mathbf{e}^{ij}$, connects the leaving point on orbit O^i to the entry point on orbit O^j. To construct $\mathbf{e}^{ij}$, we use the RRT (Rapidly exploring Random Tree) approach [21]. However, we inject user information and guide the sampling procedure in RRT to obtain better and faster results. The details of this implementation can be found in [1]. It is worth noting that in our PNPRM construction for both 2D and 3D systems, we assume that orbits are counter-clockwise in direction. An alternate approach with both clockwise and counter-clockwise orbits could also be adopted since our method is not restrictive in that sense. In this simulations, we limit ourselves to a single orbit direction for reasons of simplicity and clarity.

Planning for 6D aircraft with FIRM: After generating the PNPRM, we leverage the orbits and edges to belief space as discussed in Section V. Accordingly, we compute the edge costs and solve the DP on the FIRM graph to get a feedback from graph nodes to graph edges. Fig. 6 depicts a 3-D environment with the constructed PNPRM. The robot is given a task to visit nodes 2, 3 and 7 in that order starting from node 1. These nodes represent locations where the robot is to perform intelligence gathering. Fig. 7 shows the feedback π^g on the FIRM graph; i.e., it shows the best edge that π^g selects at each node. Shortest path is shown in green whereas the most likely path under the policy is depicted in red. It can be seen that the path selected through FIRM takes routes which are more informative and thus have less filtering uncertainty. It is worth noting that the green edges are not a part of feedback; they are just drawn to illustrate the shortest path. Fig. 8 shows the feedback to go to node 3, resulting from online replanning after the query to node 3 is submitted. Finally, Fig. 9 shows the feedback to node 7 after the next online replanning. To perform replanning (recomputing the feedback), we do not need to re-construct the graph or recompute the edge cost. Multiple queries can be executed by simply re-solving the DP on the FIRM graph with a new goal.

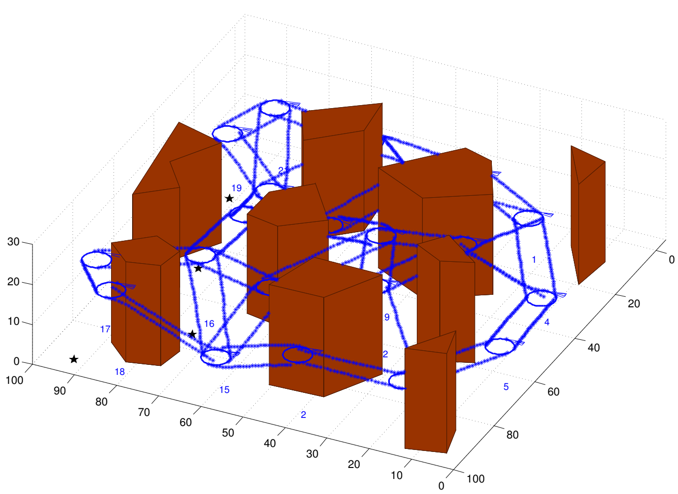
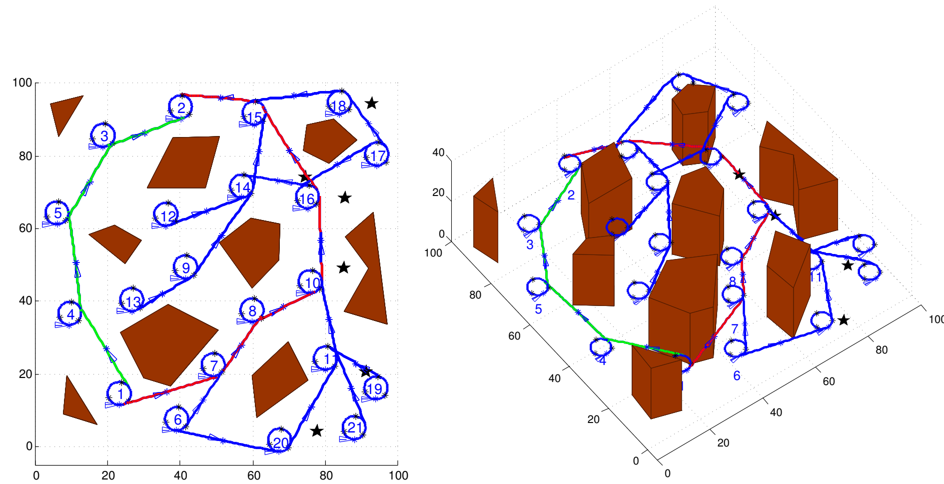


Fig. 6. The PNPRM in 3D showing the orbits and edges.

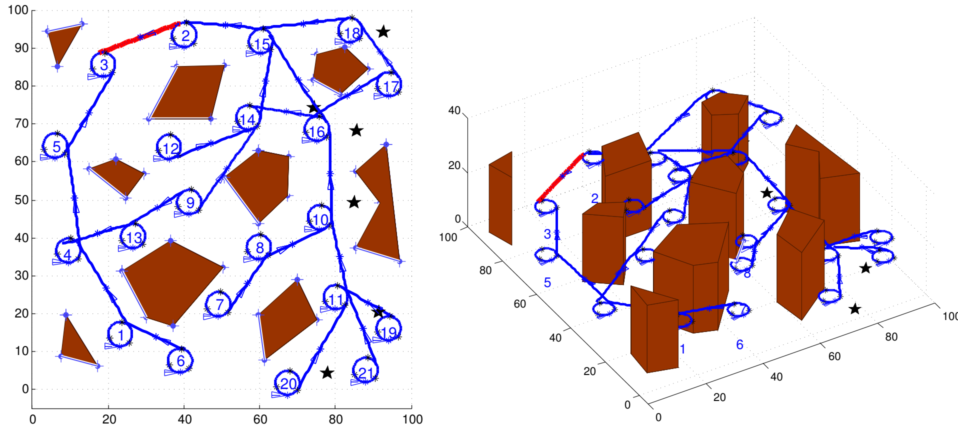
VII. Conclusion

This paper proposes a solution to the problem of stochastic planning for non-stoppable (and possibly nonholonomic) systems, such as aircraft. The Periodic-Node PRM (PNPRM) is introduced as graph in the state space, whose nodes lie on



(a) (b)

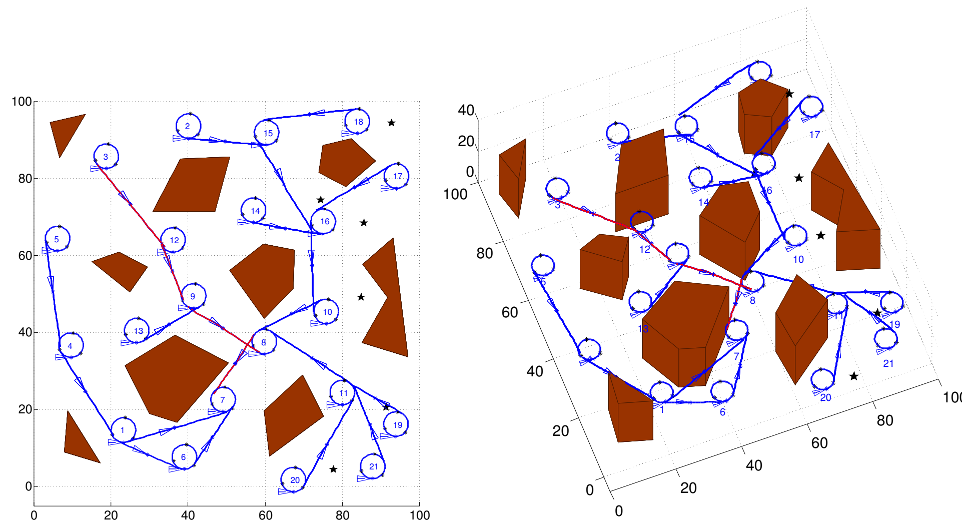
Fig. 7. Feedback π^g is shown with orbit 2 as the goal orbit. (a) Starting from orbit 1, the shortest path (green) and the most-likely path (red) are shown from top view (b) The shortest path (green) and the most-likely path (red) are shown in the 3D environment.



(a) (b)

Fig. 8. Feedback π^g is shown with orbit 3 as the goal orbit. Starting from orbit 2, the most-likely path (red) is shown (a) from top view and (b) in the 3D environment.

periodic trajectories, called orbits. Exploiting the properties of periodic LQG controllers on the orbits, we designed appropriate local controllers to accomplish the task of belief reachability for non-stoppable systems. Accordingly, by suitably choosing the belief node regions along the orbits we constructed a graph in belief space. Planning constraints can be seamlessly embedded along the edges of this graph. Finally, the framework characterizes the success probability of reaching the goal point from any given graph node. With estimation uncertainty chosen as the planning cost, simulation results for two different types of robots were presented. It was demonstrated that the proposed graph-based scheme for planning under uncertainty tends to find feedback laws that guide the robot toward goal through information-rich regions (leading to less estimation uncertainty) and regions with less collision probability.



(a) (b)

Fig. 9. Feedback π^g is shown with orbit 7 as the goal orbit. Starting from orbit 3, the most-likely path (red) is shown (a) from top view and (b) in the 3D environment.

References

[1] Aliakbar Agha-mohammadi, Saurav Agarwal, and Suman Chakravorty. Periodic-node graph-based framework for stochastic control of small aerial vehicles. Technical report, Department of Aerospace Engineering, Texas A&M University, 2014.

[2] Aliakbar Agha-mohammadi, Suman Chakravorty, and Nancy Amato. FIRM: Feedback controller-based Information-state RoadMap -a framework for motion planning under uncertainty-. In *International Conference on Intelligent Robots and Systems (IROS)*, SanFrancisco, 2011.

[3] Aliakbar Agha-mohammadi, Suman Chakravorty, and Nancy Amato. On the probabilistic completeness of the sampling-based feedback motion planners in belief space. In *IEEE International Conference on Robotics and Automation (ICRA)*, Minneapolis, 2012.

[4] Aliakbar Agha-mohammadi, Suman Chakravorty, and Nancy Amato. FIRM: Sampling-based feedback motion planning under motion uncertainty and imperfect measurements. *International Journal of Robotics Research (IJRR)*, 33(2):268–304, 2014.

[5] N. Amato, B. Bayazit, L. Dale, C. Jones, and D. Vallejo. OBPRM: An Obstacle-Basaed PRM for 3D workspaces. In *International Workshop on the Algorithmic Foundations of Robotics*, pages 155–168, 1998.

[6] Ross P. Anderson, Efstathios Bakolas, Dejan Milutinović, , and Panagiotis Tsiotras. Optimal feedback guidance of a small aerial vehicle in a stochastic wind. *Journal of Guidance, Control, and Dynamics*, 36:975–985, 2013.

[7] K Astrom. Optimal control of Markov decision processes with incomplete state estimation. *Journal of Mathematical Analysis and Applications*, 10:174–205, 1965.

[8] Dimitri Bertsekas. *Dynamic Programming and Optimal Control: 3rd Ed.* Athena Scientific, 2007.

[9] S. Bittanti, P. Bolzern, G. De Nicolao, and L. Piroddi. Representation, prediction and identification of cyclostationary processes. Cyclostationarity in Communications and Signal Processing, *Edited by W.A. Gardner, IEEE Press*, 1994.

[10] S. Bittanti and P. Colaneri. *Periodic systems filtering and control*. Springer-Verlag, 2009.

[11] Roger W. Brockett. Asymptotic stability and feedback stabilization. In R. S. Millman R. W. Brockett and H. J. Sussmann, editors, *Differential Geometric Control Theory*, pages 181–191. Birkhauser, Boston, 1983.

[12] P. Colaneri, R. Scattolini, and N. Schiavoni. LQG optimal control for multirate samled-data systems. *IEEE Transactions on Automatic Control*, 37(5):675–682, 1992.

[13] John Crassidis and John Junkins. *Optimal Estimation of Dynamic Systems*. Chapman & Hall/CRC, 2004.

[14] A. Doucet, J.F.G. de Freitas, and N.J. Gordon. *Sequential Monte Carlo methods in practice*. New York: Springer, 2001.

[15] D. Hsu. *Randomized single-query motion planning in expansive spaces*. PhD thesis, Department of Computer Science, Stanford University, Stanford, CA, 2000.

[16] V. Huynh and N. Roy. icLQG: combining local and global optimization for control in information space. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2009.

[17] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101:99–134, 1998.

[18] Sertac Karaman and Emilio Frazzoli. Sampling-based algorithms for optimal motion planning. *International Journal of Robotics Research*, 30(7):846–894, June 2011.

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

[20] P. R. Kumar and P. P. Varaiya. *Stochastic Systems: Estimation, Identification, and Adaptive Control*. Prentice-Hall, Englewood Cliffs, NJ, 1986.

[21] Steve Lavalle and J. Kuffner. Randomized kinodynamic planning. *International Journal of Robotics Research*, 20(378-400), 2001.

[22] Sean Meyn and Richard L. Tweedie. *Markov Chains and Stochastic Stability: 2nd Edition*. Cambridge University Press, 2009.

[23] Sam Prentice and Nicholas Roy. The belief roadmap: Efficient planning in belief space by factoring the covariance. *International Journal of Robotics Research*, 28(11-12), October 2009.

[24] Dan Simon. *Optimal State Estimation: Kalman, H-infinity, and Nonlinear Approaches*. John Wiley and Sons, Inc, 2006.

[25] Richard Sutton, Doina Precup, and Satinder Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112:181–211, 1999.

[26] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics*. MIT Press, 2005.

[27] Jur van den Berg, Pieter Abbeel, and Ken Goldberg. LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information. *IJRR*, 30(7):895–913, 2011.

APPENDIX I
PERIODIC LQG CONTROLLER

Periodic Linear Quadratic Gaussian (PLQG) controller is a time-varying LQG controller that is designed to track a periodic nominal trajectory in the presence of process and observation noise.

In this section, we first discuss the system linearization and nominal trajectory, and then discuss the KF, LQR and LQG designed along this trajectory. Consider the nonlinear partially-observable state-space equations of the system as follows:

$$x_{k+1} = f(x_k, u_k, w_k), \quad w_k \sim \mathcal{N}(0, Q_k) \tag{35a}$$

$$z_k = h(x_k, v_k), \quad v_k \sim \mathcal{N}(0, R_k) \tag{35b}$$

A T-periodic nominal trajectory for the robot is a sequence of planned states $(x_k^p)_{k\geq 0}$ and planned controls $(u_k^p)_{k\geq 0}$, such that it is consistent with the noiseless dynamics model, i.e., we have:

$$x_{k+1}^p = f(x_k^p, u_k^p, 0), \quad x_{k+T}^p = x_k^p, \quad u_{k+T}^p = u_k^p \tag{36}$$

The role of a closed-loop stochastic controller in tracking a nominal trajectory is to compensate for robot's deviations (due to the noise) from the nominal trajectory and to keep the robot close to the nominal trajectory in the sense of minimizing the following quadratic cost:

$$J = \mathbb{E}\left[\sum_{k\geq 0}(x_k - x_k^p)^T W_x (x_k - x_k^p) + (u_k - u_k^p)^T W_u (u_k - u_k^p)\right] \tag{37}$$

where W_x and W_u are positive definite weight matrices for state and control cost, respectively.

Since the system's state is only partially observable, at every step of LQG execution, a Kalman filter estimates the system's state and an LQR controller generates the optimal control based on this estimation. We first linearize the system along the nominal trajectory and then describe the KF and LQR designed along this path.

Model linearization: Given a periodic nominal trajectory $(x_k^p, u_k^p)_{k\geq 0}$, we linearize the dynamics and observation model in (35), as follows:

$$x_{k+1} = f(x_k^p, u_k^p, 0) + A_k(x_k - x_k^p) + B_k(u_k - u_k^p) + G_k w_k, \quad w_k \sim \mathcal{N}(0, Q_k) \tag{38a}$$

$$z_k = h(x_k^p, 0) + H_k(x_k - x_k^p) + M_k v_k, \quad v_k \sim \mathcal{N}(0, R_k) \tag{38b}$$

where

$$\begin{cases} A_k &= \frac{\partial f}{\partial x}(x_k^p, u_k^p, 0), \quad B_k = \frac{\partial f}{\partial u}(x_k^p, u_k^p, 0), \quad G_k = \frac{\partial f}{\partial w}(x_k^p, u_k^p, 0), \\ H_k &= \frac{\partial h}{\partial x}(x_k^p, 0), \quad M_k = \frac{\partial h}{\partial v}(x_k^p, 0) \end{cases} \tag{39}$$

It is worth noting that the linearized system is a periodic one, i.e.,

$$A_{k+T} = A_k, \quad B_{k+T} = B_k, \quad G_{k+T} = G_k, \quad H_{k+T} = H_k, \quad M_{k+T} = M_k, \quad Q_{k+T} = Q_k, \quad R_{k+T} = R_k. \tag{40}$$

Error system: Now, let us define the following errors:

- LQG error (main error): $e_k = x_k - x_k^p$
- KF error (estimation error): $\widetilde{e}_k = x_k - \widehat{x}_k^+$
- LQR error (mean of estimation of LQG error): $\widehat{e}_k^+ = \widehat{x}_k^+ - x_k^p$

Note that these errors are linearly dependent: $e_k = \widehat{e}_k^+ + \widetilde{e}_k$. Also, defining $\delta u_k = u_k - u_k^p$ and $\delta z_k = z_k - z_k^p := z_k - h(x_k^p, 0)$, we can rewrite above linearized models as follows:

$$e_{k+1} = A_k e_k + B_k \delta u_k + G_k w_k, \quad w_k \sim \mathcal{N}(0, Q_k) \tag{41a}$$

$$\delta z_k = H_k e_k + M_k v_k, \quad v_k \sim \mathcal{N}(0, R_k) \tag{41b}$$

which is a periodic linear system due to (40).

Periodic Kalman filter: Periodic Kalman Filter (PKF) is a time-varying Kalman filter, whose underlying linear system is periodic. In Kalman filtering, we aim to provide an estimate of the system's state based on the observations we have obtained and the control signals we have applied up to time k, i.e., $z_{0:k}$ and $u_{0,k-1}$. The estimated state is a random vector denoted by x_k^+, whose distribution is the conditional distribution of the state on the obtained data so far, which is referred to as belief and is denoted by b_k:

$$b_k = p(x_k^+) = p(x_k | z_{0:k}, u_{0:k-1}) \tag{42}$$

$$\widehat{x}_k^+ = \mathbb{E}[x_k | z_{0:k}, u_{0:k-1}] \tag{43}$$

$$P_k = \mathbb{C}[x_k | z_{0:k}, u_{0:k-1}] \tag{44}$$

where $\mathbb{E}[\cdot|\cdot]$ and $\mathbb{C}[\cdot|\cdot]$ are the conditional expectation and conditional covariance operators, respectively. In the Gaussian case, we have $b_k = \mathcal{N}(\widehat{x}_k^+, P_k)$, i.e., the belief can only be characterized by its mean and covariance. Hence, we can show b_k as the mean-covariance pair $b_k \equiv (\widehat{x}_k^+, P_k)$. Similar to the conventional Kalman filtering, PKF consists of two steps at every time stage: the prediction step and the update step. In the prediction step, the mean and covariance of prior x_k^- is computed. For the system in (41) the prediction step is:

$$\widehat{e}_{k+1}^- = A_k \widehat{e}_k^+ + B_k \delta u_k \tag{45}$$

$$P_{k+1}^- = A_k P_k^+ A_k^T + G_k Q_k G_k^T \tag{46}$$

In the update step, the mean and covariance of posterior x_k^+ is computed. For the system in (41), the update step is:

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + M_k R_k M_k^T)^{-1} \tag{47}$$

$$\widehat{e}_{k+1}^+ = \widehat{e}_{k+1}^- + K_{k+1}(\delta z_{k+1} - H_{k+1}\widehat{e}_{k+1}^-) \tag{48}$$

$$P_{k+1}^+ = (I - K_{k+1}H_{k+1})P_{k+1}^- \tag{49}$$

Note that

$$\widehat{x}_k^+ = \mathbb{E}[x_k|z_{0:k}, u_{0:k-1}] = x_k^p + \mathbb{E}[e_k|z_{0:k}, u_{0:k-1}] = x_k^p + \widehat{e}_k^+ \tag{50}$$

$$P_k = \mathbb{C}[x_k|z_{0:k}, u_{0:k-1}] = \mathbb{C}[e_k|z_{0:k}, u_{0:k-1}] = P_k^+ \tag{51}$$

Lemma 4: **(Covariance convergence under PLQG):** In Periodic Kalman filtering, if for all k, the pair (A_k, H_k) is detectable and the pair $(A_k, \check{Q}_k)$ is stabilizable, where $G_k Q_k G_k^T = \check{Q}_k \check{Q}_k^T$, then the prior covariance P_k^-, the posterior covariance P_k, and the filter gain K_k all converge to their T-periodic stationary values, denoted by $\check{P}_t^-$, $\check{P}_t$, and $\check{K}_t$, respectively [10]. Matrix $\check{P}_t^-$ is the unique Symmetric T-Periodic Positive Semi-definite (SPPS) solution [10] of the following Discrete Periodic Riccati Equation (DPRE):

$$\check{P}_{k+1}^- = G_k Q_k G_k^T + A_k(\check{P}_k^- - \check{P}_k^- H_k^T (H_k \check{P}_k^- H_k^T + M_k R_k M_k^T)^{-1} H_k \check{P}_k^-)A_k^T \tag{52}$$

Having $\check{P}_k^-$, the periodic gain $\check{K}_k$ and estimation covariance $\check{P}_k$ are computed as follows:

$$\check{K}_k = \check{P}_k^- H_k^T (H_k \check{P}_k^- H_k^T + M_k R_k M_k^T)^{-1}, \tag{53}$$

$$\check{P}_k = (I - \check{K}_k H_k)\check{P}_k^- \tag{54}$$

where

$$\check{P}_{k+T}^- = \check{P}_k^-, \quad \check{K}_{k+T} = \check{K}_k, \quad \check{P}_{k+T} = \check{P}_k \tag{55}$$

Proof: See [10]. ■

Note that if the pair (A_k, H_k) is detectable and the pair $(A_k, \check{Q}_k)$ is stabilizable, then the pair (A_k, H_k) is observable and the pair $(A_k, \check{Q}_k)$ is controllable, and hence Lemma 2 follows.

Periodic LQR controller: An LQR controller is utilized as the separated controller [20] within the structure of the LQG controller. Once Kalman filter produces the estimation (belief), the LQR controller generates the optimal control signal accordingly. In other words, we have a time-varying mapping μ_k from belief space into the control space that generates an optimal control based on the given belief $u_k = \mu_k(b_k)$ at every time step k. In LQG, the mapping μ_k is the control law of the LQR controller, which is optimal in the sense of minimizing the following cost:

$$\begin{aligned} J_{PLQR} &= \mathbb{E}\left[\sum_{k\geq 0} (\widehat{x}_k^+ - x_k^p)^T W_x (\widehat{x}_k^+ - x_k^p) + (u_k - u_k^p)^T W_u (u_k - u_k^p)\right] \\ &= \mathbb{E}\left[\sum_{k\geq 0} (\widehat{e}_k^+)^T W_x (\widehat{e}_k^+) + (\delta u_k)^T W_u (\delta u_k)\right]. \end{aligned} \tag{56}$$

The linear control law that minimizes this cost function for a linear system is:

$$\delta u_k = -L_k \widehat{e}_k^+, \quad L_{k+T} = L_k \tag{57}$$

Lemma 5: In Periodic LQR (PLQR), if for all k, the pair (A_k, B_k) is stabilizable and the pair $(A_k, \check{W}_x)$ is detectable, where $W_x = \check{W}_x^T \check{W}_x$, then the time-varying feedback gains L_k is a T-periodic gain, i.e., $L_{k+T} = L_k$ and is computed as follows:

$$L_k = (B_k^T S_{k+1} B_k + W_u)^{-1} B_k^T S_{k+1} A_k \tag{58}$$

where S_k is the SPPS solution of the following DPRE:

$$S_k = W_x + A_k^T S_{k+1} A_k - A_k^T S_{k+1} B_k (B_k^T S_{k+1} B_k + W_u)^{-1} B_k^T S_{k+1} A_k \tag{59}$$

Note that the whole control is $u_k = u_k^p + \delta u_k$.

Periodic LQG controller: Plugging the obtained control law of PLQR into the PKF equations, we can get the following error dynamics, for the defined errors:

$$\begin{aligned}\begin{pmatrix} e_{k+1} \\ \widetilde{e}_{k+1} \end{pmatrix} &= \begin{pmatrix} A_k - B_k L_k & B_k L_k \\ 0 & A_k - \check{K}_{k+1} H_{k+1} A_k \end{pmatrix} \begin{pmatrix} e_k \\ \widetilde{e}_k \end{pmatrix} \\ &+ \begin{pmatrix} G_k & 0 \\ G_k - \check{K}_{k+1} H_{k+1} G_k & -\check{K}_{k+1} M_{k+1} \end{pmatrix} \begin{pmatrix} w_k \\ v_{k+1} \end{pmatrix},\end{aligned} \tag{60}$$

or equivalently,

$$\begin{aligned}\begin{pmatrix} e_{k+1} \\ \widehat{e}^+_{k+1} \end{pmatrix} &= \begin{pmatrix} A_k & -B_k L_k \\ \check{K}_{k+1} H_{k+1} A_k & A_k - B_k L_k - \check{K}_{k+1} H_{k+1} A_k \end{pmatrix} \begin{pmatrix} e_k \\ \widehat{e}^+_k \end{pmatrix} \\ &+ \begin{pmatrix} G_k & 0 \\ \check{K}_{k+1} H_{k+1} G_k & \check{K}_{k+1} M_{k+1} \end{pmatrix} \begin{pmatrix} w_k \\ v_{k+1} \end{pmatrix}.\end{aligned} \tag{61}$$

Defining $\zeta_k := (e_k, \widehat{e}^+_k)^T$ and $q_k := (w_k, v_{k+1})^T$, we can rewrite (61) in a more compact form as

$$\zeta_{k+1} = \overline{F}_k \zeta_k - \overline{G}_k q_k, \quad q_k \sim \mathcal{N}(0, \overline{Q}_k), \quad \overline{Q}_k = \begin{pmatrix} Q_k & 0 \\ 0 & R_{k+1} \end{pmatrix} \tag{62}$$

with appropriate definitions for $\overline{F}_k$ and $\overline{G}_k$. Thus, ζ_k is a random variable with a Gaussian distribution, i.e.,

$$\zeta_k \sim \mathcal{N}(0, \mathcal{P}_k), \tag{63}$$

or

$$\begin{pmatrix} x_k \\ \widehat{x}^+_k \end{pmatrix} \sim \mathcal{N}(\begin{pmatrix} x^p_k \\ x^p_k \end{pmatrix}, \mathcal{P}_k), \tag{64}$$

where $\mathcal{P}_k$ is the solution of the following Discrete Periodic Lyapunov Equation (DPLE):

$$\mathcal{P}_{k+1} = \overline{F}_k \mathcal{P}_k \overline{F}_k^T - \overline{G}_k \overline{Q}_k \overline{G}_k^T, \tag{65}$$

which can be decomposed into four blocks

$$\mathcal{P}_k = \begin{pmatrix} \mathcal{P}_{k,11} & \mathcal{P}_{k,12} \\ \mathcal{P}_{k,21} & \mathcal{P}_{k,22} \end{pmatrix}. \tag{66}$$

Lemma 6: Under the preceding assumptions in Lemmas 4 and 5, the solution of DPLE in (65) converges to a unique SPPS solution $\check{\mathcal{P}}_k$ independent of the initial covariance $\mathcal{P}_0$, i.e., $\check{\mathcal{P}}_{k+T} = \check{\mathcal{P}}_k$.

Proof: See [10]. ■

Therefore, the process in (62) converges to a cyclostationary process [9], i.e., the distribution over ζ_k is periodic. Thus, since $\widehat{x}^+_k \sim \mathcal{N}(x^p_k, \mathcal{P}_{k,22})$, the distribution over estimation mean is also converges to a periodic distribution, i.e., $\widehat{x}^+_k \sim \mathcal{N}(x^p_k, \check{\mathcal{P}}_{k,22}) = \mathcal{N}(x^p_{k+T}, \check{\mathcal{P}}_{k+T,22})$. Hence, this analysis leads to the following lemma:

Lemma 7: Under Periodic LQG, belief falls into a Gaussian cyclostationary process, i.e., the distribution over belief $b_k = (\widehat{x}^+_k, P_k)$ converges to the following periodic Gaussian distribution:

$$b_k \equiv (\widehat{x}^+_k, P_k) \sim \mathcal{N}\left(\begin{pmatrix} x^p_k \\ \check{P}_k \end{pmatrix}, \begin{pmatrix} \check{\mathcal{P}}_{k,22} & 0 \\ 0 & 0 \end{pmatrix}\right) \tag{67}$$

The degeneracy of the Gaussian distribution over belief in (67) is due to the fact that $\check{P}_k$ is a deterministic process. It is worth noting that the belief mean converges to the T-periodic belief $\mathbb{E}[b_{k+T}] = \mathbb{E}[b_k] = (x^p_k, \check{P}_k)$. Hence, the Lemma 1 follows, as it is the same as Lemma 7, where we have:

$$b^c_k = \begin{pmatrix} x^p_k \\ \check{P}_k \end{pmatrix}, \quad \mathtt{C}_k = \begin{pmatrix} \check{\mathcal{P}}_{k,22} & 0 \\ 0 & 0 \end{pmatrix} \tag{68}$$

APPENDIX II
PROOF OF LEMMA 3

Proof: Let us consider the state space model of a T-periodic linear system of interest as follows:

$$x_{k+1}= \mathbf{A}_k x_k + \mathbf{B}_k u_k + \mathbf{G}_k w_k, \quad w_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_k) \tag{69a}$$

$$z_k= \mathbf{H}_k x_k + v_k, \qquad v_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k). \tag{69b}$$

Based on Lemma 1 and Lemma 2, if $(\mathbf{A}, \mathbf{B})$ and $(\mathbf{A}, \check{\mathbf{Q}})$ are controllable pairs, where $\mathbf{G}\mathbf{Q}\mathbf{G}^T = \check{\mathbf{Q}}\check{\mathbf{Q}}^T$, and if $(\mathbf{A}, \mathbf{H})$ and $(\mathbf{A}, \check{\mathbf{W}}_x)$ are observable pairs, where $\mathbf{W}_x = \check{\mathbf{W}}_x^T \check{\mathbf{W}}_x$, then the estimation covariance deterministically tends to a T-periodic stationary covariance $\check{P}_k$. Therefore, for any $\epsilon > 0$, after a deterministic finite time, P_k enters the ϵ-neighborhood of the periodic stationary covariance, i.e., $\|P_k - \check{P}_k\|_m < \epsilon$ for all k large enough, where $\| \cdot \|$ stands for a matrix norm.

The estimation mean dynamics, however, is stochastic and is as follows for the system in Eq. (69):

$$\begin{aligned}
\widehat{x}^+_{k+1} &= x^p_{k+1} + (\mathbf{A}_k - \mathbf{B}_k \mathbf{L}_k - \mathbf{K}_{k+1}\mathbf{H}_{k+1}\mathbf{A}_k)(\widehat{x}^+_k - x^p_k) \\
&+ \mathbf{K}_{k+1}\mathbf{H}_{k+1}\mathbf{A}_k(x_k - x^p_k) + \mathbf{K}_{k+1}\mathbf{H}_{k+1}\mathbf{G}_k w_k + \mathbf{K}_{k+1} v_{k+1} \\
&= x^p_{k+1} - (\mathbf{A}_k - \mathbf{B}_k \mathbf{L}_k) x^p_k + (\mathbf{A}_k - \mathbf{B}_k \mathbf{L}_k - \mathbf{K}_{k+1}\mathbf{H}_{k+1}\mathbf{A}_k)\widehat{x}^+_k \\
&+ \mathbf{K}_{k+1}\mathbf{H}_{k+1}\mathbf{A}_k x_k + \mathbf{K}_{k+1}\mathbf{H}_{k+1}\mathbf{G}_k w_k + \mathbf{K}_{k+1} v_{k+1}
\end{aligned} \tag{70}$$

where the Kalman gain $\mathbf{K}_k$ is:

$$\mathbf{K}_k = P^-_k \mathbf{H}^T_k (\mathbf{H}_k P^-_k \mathbf{H}^T_k + \mathbf{R}_k)^{-1} \tag{71}$$

Since $\mathbf{K}_k$ is full rank (due to the condition on the rank of $\mathbf{H}_k$) for all k and since v and w are Gaussian noises, (70) induces an irreducible Markov process over the state space [22]. Thus, if we have a stopping region for the estimation mean with size $\epsilon > 0$, the estimation mean process will hit this stopping region in finite time [22], with probability one, i.e., for a finite $\mathbf{v} \in \mathbb{X}$, the condition $\|\widehat{x}^+_k - \mathbf{v}\| < \epsilon$ is satisfied in finite time. However, $\mathbf{v}$ can be chosen in a way that maximizes the absorption probability and minimizes the hitting time.

Based on the estimation mean dynamics in (70) and the state dynamics in Appendix I, if the estimation mean process and state process start from $\widehat{x}^+_0$ and x_0, respectively, such that $\mathbb{E}[\widehat{x}^+_0] = x^p_k$ and $\mathbb{E}[x_0] = x^p_k$ (which indeed is the case in FIRM due to the usage of edge-controllers), "the mean of estimation mean" remains on x^p_k, i.e., $\mathbb{E}[\widehat{x}^+_k] = x^p_k$, for all k. As a result, x^p_k is the optimal choice for the center of stopping region and thus, the condition $\|\widehat{x}^+_k - x^p_k\| < \epsilon$ is satisfied in a minimum time in the sense of "expected value".

Combining the results for estimation covariance and estimation mean, if we define the region $\check{B}_k$ as a set in the Gaussian belief space with a non-empty interior centered at $(x^p_{k_\alpha}, \check{P}_{k_\alpha})$, then the belief $b_k = (\widehat{x}^+_k, P_k)$ enters region $\cup_k \check{B}_k$ with minimum finite expected time with probability one. To decrease the number of nodes, one can only look at the subsequence $\grave{b}_\alpha := b_{k_\alpha} = (\widehat{x}^+_{k_\alpha}, P_{k_\alpha})$ and $B_\alpha := \check{B}_{k_\alpha}$ for $\{k_1, k_2, \cdots, k_m\} \subset \{1, 2, \cdots, T\}$, then similarly the belief $\grave{b}_\alpha$ enters region $\cup_\alpha B_\alpha$ in finite time with probability one. ■