A Framework for Constrained Deployment Optimization of Wireless Mobile Sensor Networks

Mohammadreza Barzegaran and Hamid Jafarkhani

Center for Pervasive Communications and Computing, University of California, Irvine, California 92697

Email: {barzegm1, hamidj}@uci.edu

Abstract—The expansion of mobile sensors, like robots and un-crewed aerial vehicles (UAVs), across diverse applications such as remote sensing, monitoring, and communication relay, has been exponential. Yet, ensuring their safe and successful operation depends crucially on optimized deployment tailored to the application requirements while constrained by various limitations. This study focuses on the optimization of robot/UAV trajectories under these constraints. However, implementing constraints poses considerable challenges. To this end, a framework for constrained deployment optimization of wireless robotic swarms is proposed. This framework formulates as a quadratic-programming problem which utilizes Bézier curves to model trajectories and predict their states over a time horizon. Constraints are systematically categorized and embedded in the Bézier curve formulation. This framework offers ease of adoption to various scenarios and flexibility in accommodating different mobile sensor dynamics, constraints, and deployment strategies.

Index Terms—Swarm deployment, Wireless mobile sensor networks, Uncrewed aerial vehicle (UAV), Predictive control

I. INTRODUCTION

Gathering high-accuracy data, repeatability, predictability, reliability, flexibility, shorter time-to-deployment, and lower costs are among the key factors which make mobile sensors, like robots and uncrewed aerial vehicles (UAVs), the best choice in diverse applications such as remote sensing, monitoring, and connectivity. However, these benefits come with various challenges. These challenges become more complex when deploying a swarm of heterogeneous mobile sensors, or implementing robot's non-linear dynamics, or shifting from two-dimensional (2D) to three-dimensional (3D) deployment, for example deploying UAVs. These challenges originate from the constraints imposed by the mobile sensors themselves, the deployment environment, or the application requirements. Various constraints have been defined including energy constraints [1]–[4], connectivity constraints [5]–[9], collision constraints [10], [11], obstacle constraints [11], [12], regulatory constraints [13], maneuverability constraints [14], actuator constraints [15], among others. Similarly, various solutions have been proposed to optimize robot deployment while satisfying these constraints [1], [3], [7], [8], [10], [11]. However these solutions are tailored to specific applications and constraints, requiring significant effort for adopting to other applications and constraints.

In this paper, we address the problem of optimizing the deployment of a swarm of mobile sensors under diverse characteristics and while satisfying various constraints. Such optimized deployment can be synthesized in design-time (when requirements, constraints, and targets are pre-defined) and real-time (when decision is made based on the available information and context). In scenarios where deployment targets are not pre-defined, various methods such as gradient approaches [16], greedy algorithms [17], and particle swarm optimization [18], might be employed to search and find the optimal deployment.

We propose a framework that easily adopts diverse constraints that (*i*) relate to the derivatives of robot positions, (*ii*) require repulsion, (*iii*) require attraction, and (*iv*) require arrival. To this end, we represent the robot/UAV trajectories with Bézier curves [19], which are parametric curves. Adjusting the curve parameters defines the robot/UAV trajectory. Although other parametric curve formulations exist in the literature, the unique formulation of Bézier curves offer several advantages: first, they facilitate the prediction of robot states over a time horizon by sampling the curves as inputs to the mobile sensor's dynamics; second, constraints can be imposed directly on the curve geometry, embedding them into the curve formulation.

The contributions of this paper are as follows:

- We address the deployment of robots including mobile vehicles in a 2D space and UAVs in a 3D space, proposing a framework that easily adopt to diverse requirements.
- Our novel framework can be utilized for both design-time and real-time decision making, and adopts with time-varying networks.
- We employ Bézier curves for modeling mobile sensor's trajectories and predict their behavior over a time horizon.
- We identified four categories of deployment constraints and embedded them into Bézier curves formulation.

The remainder of the manuscript is organized as follows: The organization of the paper is as follows. The formal problem formulation is presented in Sec. II. Sec. III introduces Bézier curves, a core component of our framework, and provides essential information about them. Our novel framework, which categorizes and embeds constraints into the Bézier curve formulation, is detailed in Sec. IV. We demonstrate our framework on a test case and present key indicators in Sec. V. Concluding remarks and future work are discussed in Sec. VI.

II. PROBLEM FORMULATION

The problem addressed in this paper is general and encompasses a variety of applications where robots and UAVs are deployed to optimize performance indicators. Given their diverse requirements and constraints, we present our flexible

framework that accommodates these various constraints and can implement various search algorithms. The formal problem formulation is provided below:

This paper considers the deployment of a swarm of N heterogeneous mobile sensors in a bounded space $\boldsymbol{\Omega}$, which can be either 2D or 3D. A point in the space is denoted with a vector $\boldsymbol{\omega} \in \boldsymbol{\Omega}$. The swarm is denoted with $\mathcal{N} = \{n | 1 \leq n \leq N\}$, where each index n represents a mobile sensor (a robot in 2D or a UAV in 3D).

Robot n's tracking dynamics is defined as:

$$\dot{\boldsymbol{p}}_n(t) = f\big(\boldsymbol{p}_n(t)\big) + g\big(\boldsymbol{u}_n(t)\big), \tag{1}$$

where $\boldsymbol{p}_n(t) \in \boldsymbol{\Omega}$ and $\boldsymbol{u}_n(t) \in \boldsymbol{\Omega}$ represent its position and its desired position, respectively. The robot tracking dynamics can accommodate simple integrator, uni-cycle, random waypoint mobility (RWM), state-space, or control affine form.

Let us denote $\check{\boldsymbol{p}}_n(t)$ as the target position that achieves the optimal deployment. To reach the target positions, we consider each robot traverses on a trajectory whose points represent $\boldsymbol{u}_n(t)$ in (1). This trajectory is defined as a Bézier curve. Thus, by adjusting the Bézier curve parameters, a trajectory is defined that enables the robot to reach its target position when followed.

As mentioned earlier, various constraints have been defined such as (*i*) Domain constraints: The deployment space may include obstacles, restricted airspace, or other similar constraints; (*ii*) Collision avoidance: Robots must avoid collision; (*iii*) Connectivity: Robots must communicate with each other to coordinate their actions and transmit the collected data to the outside world; (*iv*) Maneuverability constraints: Robots tracking their trajectories must not attempt maneuvers that exceed their capabilities; and (*v*) Energy constraints: Each Robot's energy consumption must be within its energy capacity. The safe and successful robot deployment depends on satisfying these constraints. All these constraints can be embedded into the Bézier curve formulation by adjusting the Bézier curve parameters. This concept is presented and analyzed in the next section.

III. BACKGROUND

In geometry, a line is a set of points satisfying a linear equation: $\boldsymbol{p}(t) = \boldsymbol{p}(t_0) + k\mathbf{v}$, where $\boldsymbol{p}(t_0)$ is a point, $\mathbf{v}$ is a non-zero direction vector, and k is a scalar parameter. Alternatively, $\mathbf{v}$ can be defined as $\mathbf{v} = \boldsymbol{p}(t_0) - \boldsymbol{p}(t_1)$, where $\boldsymbol{p}(t_1)$ is an additional point. In contrast, a curve can have arbitrary shape and extension. Simple curves like circular arcs or parabolic segments are defined by three non-collinear points. Adding more points adjusts the curve and increases the complexity. This is the idea behind higher-degree polynomial curves, splines, and Bézier curves [19]. In a Bézier curve, a set of $T + 1$ discrete *control points*, with the τth control point denoted with $\boldsymbol{\lambda}^{\tau-1}$, are used in a formula to define a $T + 1$ degree smooth and continuous polynomial curve. The curve starts at $\boldsymbol{\lambda}^0$, ends at $\boldsymbol{\lambda}^T$, and stays within the convex hull formed by these control points. More control points (a higher degree curve) allow for a more complex curve. A Bézier curve

can be elevated to higher degrees with the same shape [20]. Formally, a Bézier curve is defined as [19]

$$\boldsymbol{\ell}(k) = \sum_{\tau=0}^{T} \boldsymbol{\lambda}^\tau \mathfrak{B}_\tau^T(k), \tag{2}$$

where $k \in [0, 1]$ is a scalar parameter and $\mathfrak{B}_\tau^T(k)$ denotes the Bernstein polynomials [21], defined as

$$\mathfrak{B}_\tau^T(k) = \binom{T}{\tau} (1 - k)^{T-\tau} (k)^\tau. \tag{3}$$

A Bézier curve inherits smoothness and its derivative is calculated as [21]

$$\dot{\boldsymbol{\ell}}(k) = T \sum_{\tau=0}^{T-1} \big(\boldsymbol{\lambda}^{\tau+1} - \boldsymbol{\lambda}^\tau\big) \mathfrak{B}_\tau^{T-1}(k). \tag{4}$$

Higher-order derivatives are calculated similarly. The squared Euclidean distance between the two same-degree Bézier curves $\boldsymbol{\ell}_n(k)$ and $\boldsymbol{\ell}_m(k)$ is calculated as [22]:

$$d^2(k) = ||\boldsymbol{\ell}_n(k) - \boldsymbol{\ell}_m(k)||^2 = ||\sum_{\tau=0}^{T} \boldsymbol{\lambda}_d^\tau \mathfrak{B}_\tau^T(k)||^2$$
$$= \sum_{i=0}^{T} \sum_{j=0}^{T} \big(\boldsymbol{\lambda}_d^i\big)^{\text{T}} \boldsymbol{\lambda}_d^j \frac{\binom{T}{i}\binom{T}{j}}{\binom{2T}{i+j}} \mathfrak{B}_{i+j}^{2T}(k), \tag{5}$$

since the product of the Bernstein polynomials is [22]

$$\mathfrak{B}_i^T(k)\mathfrak{B}_j^T(k) = \frac{\binom{T}{i}\binom{T}{j}}{\binom{2T}{i+j}} \mathfrak{B}_{i+j}^{2T}(k). \tag{6}$$

IV. SOLUTION

As discussed earlier, the target position $\check{\boldsymbol{p}}_n(t)$ achieves the optimal deployment, for example by maximizing the coverage. The target positions can be defined as

$$\check{\boldsymbol{p}}_n(t) = h(\boldsymbol{p}_n(t), t), \tag{7}$$

where $h(\boldsymbol{p}_n(t), t)$ implements invariant positions or methods such as gradient ascend [16].

We consider four constraint categories in this work: (*i*) derivative related constraints: Derivatives of robot trajectories are upper/lower bounded; (*ii*) minimum distance constraints: The Euclidean distance between two robots are lower bounded; (*iii*) maximum distance constraints: The Euclidean distance between two robots are upper bounded; and (*iv*) visit constraints: Robots must visit a specific point in the space.

Derivative related constraints: An example of such constraints is the robots' velocity limitations. Since robots are tracking a trajectory defined by a Bézier curve, this constraint can be applied to the first derivative of Bézier curves to derive

$$||\dot{\boldsymbol{\ell}}_n(k)|| \leq \text{V}_n(\boldsymbol{p}_n(t), t), \tag{8}$$

where $\text{V}_n(\boldsymbol{p}_n(t), t)$ represents the velocity upper bound. Similarly, constraints on robots' acceleration can be defined using higher-order derivatives of Bézier curves.

Minimum distance constraints: Throughout the deployment, robots must avoid collision, i.e., they must maintain a safe distance with each other or obstacles. Additionally, they must not enter regions where their operation is unsafe or access is limited. For a single point $\boldsymbol{\omega} \in \boldsymbol{\Omega}$, this constraint is defined

as
$$\left(\boldsymbol{\lambda}_n^i - \boldsymbol{\omega}\right)^{\mathrm{T}}\left(\boldsymbol{\lambda}_n^j - \boldsymbol{\omega}\right) \geq \mathrm{R}(\boldsymbol{p}_n(t), \boldsymbol{\omega}, t)^2, \quad 0 \leq i, j \leq T, \tag{9}$$

where $\mathrm{R}(\boldsymbol{p}_n(t), \boldsymbol{\omega}, t)$ is the safe distance between $\boldsymbol{p}_n(t)$ and $\boldsymbol{\omega}$.

Proposition 1. *A Bézier curve $\boldsymbol{\ell}(k)$ of degree T remains within a safe distance r of a point $\boldsymbol{\omega}$ if any pair of its control points $\boldsymbol{\lambda}^i$ and $\boldsymbol{\lambda}^j$ satisfy the following:*
$$\left(\boldsymbol{\lambda}^i - \boldsymbol{\omega}\right)^{\mathrm{T}}\left(\boldsymbol{\lambda}^j - \boldsymbol{\omega}\right) \geq r^2, \quad 0 \leq i, j \leq T. \tag{10}$$

Proof. Given a Bézier curve $\boldsymbol{\ell}(k)$ of degree T and control points $\boldsymbol{\lambda}$, using [23], the squared Euclidean distance between the curve and a point $\boldsymbol{\omega}$ is defined as
$$d^2(k) = \|\sum_{i=0}^{T} \boldsymbol{\lambda}^i \mathfrak{B}_i^T(k) - \boldsymbol{\omega}\|^2$$
$$= \|\sum_{i=0}^{T} \underbrace{\left(\boldsymbol{\lambda}^i - \boldsymbol{\omega}\right)}_{\boldsymbol{\lambda}_d^i} \mathfrak{B}_i^T(k)\|^2. \tag{11}$$

Using (5), we have
$$d^2(k) = \sum_{i=0}^{T}\sum_{j=0}^{T} \underbrace{\left(\boldsymbol{\lambda}_d^i\right)^{\mathrm{T}}\boldsymbol{\lambda}_d^j}_{\alpha_{i,j}} \frac{\binom{T}{i}\binom{T}{j}}{\binom{2T}{i+j}}\mathfrak{B}_{i+j}^{2T}(k). \tag{12}$$

Assuming $\alpha_{i,j} \geq r^2$, $0 \leq i, j \leq T$, as in (10), we have
$$\sum_{i=0}^{T}\sum_{j=0}^{T} \alpha_{i,j}\frac{\binom{T}{i}\binom{T}{j}}{\binom{2T}{i+j}}\mathfrak{B}_{i+j}^{2T}(k) \geq \sum_{i=0}^{T}\sum_{j=0}^{T} r^2\frac{\binom{T}{i}\binom{T}{j}}{\binom{2T}{i+j}}\mathfrak{B}_{i+j}^{2T}(k)$$
$$\sum_{i=0}^{T}\sum_{j=0}^{T} \alpha_{i,j}\frac{\binom{T}{i}\binom{T}{j}}{\binom{2T}{i+j}}\mathfrak{B}_{i+j}^{2T}(k) \geq r^2 \underbrace{\sum_{i=0}^{T}\sum_{j=0}^{T} \frac{\binom{T}{i}\binom{T}{j}}{\binom{2T}{i+j}}\mathfrak{B}_{i+j}^{2T}(k)}_{\beta}, \tag{13}$$

where $\beta = 1$ because of the Bernstein polynomials. Thus,
$$\sum_{i=0}^{T}\sum_{j=0}^{T} \alpha_{i,j}\frac{\binom{T}{i}\binom{T}{j}}{\binom{2T}{i+j}}\mathfrak{B}_{i+j}^{2T}(k) \geq r^2,$$
$$d(k) \geq r. \tag{14}$$
$\square$

Similarly, to avoid collision between Robot n and Robot m with a safe distance $\mathrm{R}(\boldsymbol{p}_n(t), \boldsymbol{p}_m(t), t)$, the corresponding constraint, assuming the same-degree Bézier curves, is:
$$\left(\boldsymbol{\lambda}_n^j - \boldsymbol{\lambda}_m^j\right)^{\mathrm{T}}\left(\boldsymbol{\lambda}_n^j - \boldsymbol{\lambda}_m^j\right) \geq \mathrm{R}(\boldsymbol{p}_n(t), \boldsymbol{p}_m(t), t)^2. \tag{15}$$
If the degrees do not match, the lower degree Bézier curve can be elevated to match the higher-degree curve.

Proposition 2. *Given two Bézier curves $\boldsymbol{\ell}(k)$ and $\boldsymbol{\ell}'(k)$ of the same degree T defined with the control points $\boldsymbol{\lambda}$ and $\boldsymbol{\lambda}'$, respectively, the Euclidean distance between the two curves is larger than or equal to r if*
$$\left(\boldsymbol{\lambda}_d^i\right)^{\mathrm{T}}\boldsymbol{\lambda}_d^j \geq r^2, \quad 0 \leq i, j \leq T, \tag{16}$$
where $\boldsymbol{\lambda}_d = \boldsymbol{\lambda} - \boldsymbol{\lambda}'$.

Proof. Using (5), the squared Euclidean distance between two Bézier curves $\boldsymbol{\ell}(k)$ and $\boldsymbol{\ell}'(k)$ with the same degree can be

expressed as
$$d^2(k) = \sum_{i=0}^{T}\sum_{j=0}^{T} \underbrace{\left(\boldsymbol{\lambda}_d^i\right)^{\mathrm{T}}\boldsymbol{\lambda}_d^j}_{\alpha_{i,j}} \frac{\binom{T}{i}\binom{T}{j}}{\binom{2T}{i+j}}\mathfrak{B}_{i+j}^{2T}(k). \tag{17}$$
The rest of the proof is very similar to that of Proposition 1 and is omitted for brevity. $\square$

Fig. 1 provides an example where the black and blue curves fail to satisfy the condition in (15) for a distance of 25, resulting in collisions indicated by the red lines. Readers are encouraged to explore this constraint further using our web application available at http://bezierwebapp.barzegaran.xyz.

Maximum distance constraints: An example of such constraints is connectivity, which requires two nodes remain within a maximum distance of each other. As a more specific example, a robot may need to maintain connections with a given base station. For a single point $\boldsymbol{\omega} \in \boldsymbol{\Omega}$ and a maximum distance $\mathrm{R}(\boldsymbol{p}_n(t), \boldsymbol{\omega}, t)$, this constraint is defined as
$$\left(\boldsymbol{\lambda}_n^i - \boldsymbol{\omega}\right)^{\mathrm{T}}\left(\boldsymbol{\lambda}_n^j - \boldsymbol{\omega}\right) \leq \mathrm{R}(\boldsymbol{p}_n(t), \boldsymbol{\omega}, t)^2, \quad 0 \leq i, j \leq T. \tag{18}$$

Proposition 3. *A Bézier curve $\boldsymbol{\ell}(k)$ of degree T maintains a maximum distance r to a point $\boldsymbol{\omega}$ if any pair of its control points $\boldsymbol{\lambda}^i$ and $\boldsymbol{\lambda}^j$ satisfy the following:*
$$\left(\boldsymbol{\lambda}^i - \boldsymbol{\omega}\right)^{\mathrm{T}}\left(\boldsymbol{\lambda}^j - \boldsymbol{\omega}\right) \leq r^2, \quad 0 \leq i, j \leq T. \tag{19}$$

Proof. The proof is identical to the proof of Proposition 1 replacing $\geq$ with $\leq$. $\square$

Alternatively, a robot may need to maintain a maximum distance to a set of points representing another robot's trajectory. The decision of which robots require connectivity can be determined using a topology control policy $\Gamma : \mathcal{N} \longrightarrow \mathcal{N}$. The policy maps Robot n to a subset of robots denoted by $\mathcal{N}_n \subset \mathcal{N}$, for example the topology control policy presented in [24]. With a maximum desired distance between Robot n and Robot $m \in \mathcal{N}_n$ as $\mathrm{R}(\boldsymbol{p}_n(t), \boldsymbol{p}_m(t), t)$, this constraint for the same-degree Bézier curves is defined as
$$\left(\boldsymbol{\lambda}_n^j - \boldsymbol{\lambda}_m^j\right)^{\mathrm{T}}\left(\boldsymbol{\lambda}_n^j - \boldsymbol{\lambda}_m^j\right) \geq \mathrm{R}(\boldsymbol{p}_n(t), \boldsymbol{p}_m(t), t)^2, \ \forall m \in \Gamma(n), \tag{20}$$
where $0 \leq i, j \leq T$.

Proposition 4. *Given two Bézier curves $\boldsymbol{\ell}(k)$ and $\boldsymbol{\ell}'(k)$ of the same degree T defined with the control points $\boldsymbol{\lambda}$ and $\boldsymbol{\lambda}'$, respectively, the Euclidean distance between the two curves is less than or equal to r if*
$$\left(\boldsymbol{\lambda}_d^i\right)^{\mathrm{T}}\boldsymbol{\lambda}_d^j \leq r^2, \quad 0 \leq i, j \leq T, \tag{21}$$

Fig. 1. Keeping the distance between two Bézier curves more than a constant.

where $\boldsymbol{\lambda}_d = \boldsymbol{\lambda} - \boldsymbol{\lambda}'$.

Proof. The proof is identical to the proof of Proposition 2 replacing $\geq$ with $\leq$. $\qquad\square$

Fig. 2 shows an example, generated by our web application, where the black and blue curves fail to satisfy the condition in (20) for a distance of 100, resulting in communication loss indicated by the orange lines.

Visit constraints: An example of such constraints is continuity in iterative implementation, which requires robot trajectories to extend from the robot positions $\boldsymbol{p}_n(t)$, ensuring continuous trajectories throughout the deployment. This constraint is defined as

$$\boldsymbol{\lambda}_n^0 = \boldsymbol{p}_n(t_0), \tag{22}$$

where t_0 is the current time.

Constraint optimization and QP: As discussed earlier, the trajectory defined by Bézier curve $\boldsymbol{\ell}_n(k)$ is the input to the robot's dynamics given in (1). We then use these dynamics to predict the robot's position, denoted as $\hat{\boldsymbol{p}}_n(t)$, over the horizon $t \in [t_0, t_f]$ as

$$\hat{\boldsymbol{p}}_n(t) = \boldsymbol{p}_n(t_0) + \int_{t_0}^{t} \big(f\big(\boldsymbol{p}_n(t)\big) + g\big(\boldsymbol{\ell}_n(\kappa t')\big)\big)dt', \tag{23}$$

where t_0 is the start of the horizon, t_f is the end of the horizon, and $\kappa = \frac{1}{t_f - t_0}$ ensures that the Bézier curve's scalar parameter remains in the range [0, 1]. Given the target position defined by (7) and the predicted positions from (23), we formulate the following quadratic programming (QP) problem to determine a trajectory that guides the robots to the target positions while enforcing the required constraints:

$$\min_{\boldsymbol{\lambda}_n} ||\hat{\boldsymbol{p}}_n(t_f) - \check{\boldsymbol{p}}_n(t)||^2,$$

$$\text{s.t.} \quad (8), (9), (15), (18), (20), (22). \tag{24}$$

Solving the QP problem in (24) provides the control points $\boldsymbol{\lambda}_n$ that define the trajectories using Bézier curves $\boldsymbol{\ell}_n(k)$. The constraint in (22) ensures that these trajectories start from the current robots' positions. Consequently, remaining at the current positions is a feasible solution. In addition, the QP problem in (24) determines trajectories where all points satisfy all constraints, and the end of trajectories $\hat{\boldsymbol{p}}_n(t_f)$, $n = 1, \cdots, N$ have the minimum distance to target positions $\check{\boldsymbol{p}}_n(t)$, $n = 1, \cdots, N$. Fig. 3 shows an example where the black curve is the result of optimization in (24), in which (S), (E), and (T), represent robot's current/starting position $\boldsymbol{p}_n(t_0)$, the end of the trajectory, i.e., final position $\hat{\boldsymbol{p}}_n(t_f)$, and target position $\check{\boldsymbol{p}}_n(t)$, respectively.
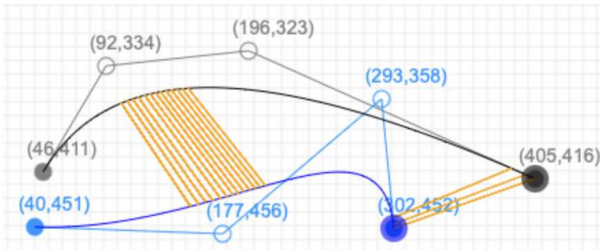

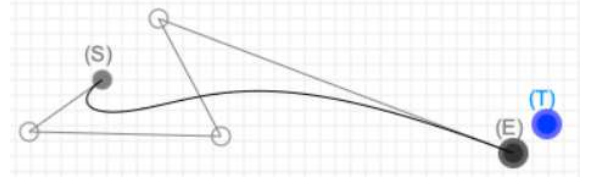Fig. 2. Keeping the distance between two Bézier curves less than a constant.

Fig. 3. Example trajectory solution.

To summarize, mobile sensors following the trajectory solutions from (24) will satisfy all imposed constraints and approach the target positions $\check{\boldsymbol{p}}_n(t)$, $n = 1, \cdots, N$. As discussed earlier, iteratively solving (24) and (7) ensures mobile sensors' positions converge to the optimal deployment.

V. Evaluation

In this section, first, we compare the characteristics of our proposed framework with those of the related work in the literature. The compared related work, as summarized in Table I, consists of: (*i*) optimal control (OC) [3], (*ii*) deep reinforcement learning (DRL) [1], (*iii*) predictive control (PC) [10], and (*iv*) barrier functions (BF) [11]. Table I's first column includes the list of characteristics for each work, separated by horizontal lines into three categories of deployment environment, constraint categories, and solution types. Our proposed framework is the only study that accommodates all four categories of constraints and can manage various environments and solution types.

We also compare the use of Bézier curves with other parametric curve formulations including B-spline, Catmull-Rom splines (CRS), Lagrange Polynomials curves (LPc), and Pythagorean Hodograph curves (HPc) [25]. The comparison consists: (*i*) Shape control mechanism, (*ii*) Simplicity, (*iii*) Smoothness and continuity, (*iv*) Starting and Ending, and (*v*) Distance computation. In terms of Shape control, Bézier curves and LPc use control points that directly influence the curve's shape, with each control point affecting the entire curve although LPc may exhibit oscillations. The control points in B-spline only influence the local shape of the curve. Meanwhile, CRS are defined by tangents at control points, affecting the curve shape through pairs of control points. Finally, HPc require additional parameters to control their shape.

In terms of simplicity, HPc have the most complex formulation, making them difficult to compute. B-splines are relatively complex, requiring knowledge of knot points. CRS and LPc are simpler, but Bézier curves offer the most straightforward formulation, making them easy to understand and implement. Furthermore, Bézier curves, B-splines, and HPc have well-defined and smooth derivatives, making them suitable for applications requiring higher-order continuity. CRS and LPc may have complex or non-smooth higher order derivatives.

Bézier curves, CRS, and LPc naturally pass through their start and end points. In contrast, B-splines and LPc do not necessarily pass through their control points, including the start and end points. In terms of distance computation, Bézier curves, HPc, and B-Splines have direct analytic solutions for distance computation in special cases, whereas CRS and LPc

Characteristics	OC [3]	DRL [1]	PC [10]	BF [11]	This work
Space dimension adaptation	2D	2D & 3D	2D & 3D	2D	2D & 3D
Robot dynamics adaptation	differential-driven	uni-cycle	state-space	any dynamics	any dynamics
Derivative related constraints	Yes	Yes	Yes	Yes	Yes
Minimum distance constraints	Yes	Yes	Yes	Yes	Yes
Maximum distance constraints	No	Yes	No	No	Yes
Visit constraints	Yes	No	No	Yes	Yes
Time domain adaptation	cont-T[1]	disc-T[2]	disc-T	cont-T	cont-T & cisc-T
Decision making adaptation	des-T[3]& real-T[4]	des-T	des-T & real-T	des-T & real-T	des-T & real-T
Target determination adaptation	static	static & dynamic	static & dynamic	static & dynamic	static & dynamic

[1] continuous time; [2] discrete time; [3] design-time; [4] real-time.

generally require numerical methods for distance calculation.

Based on this information, Bézier curves are our most suitable solution as they are simple, differentiable, and continuous. They offer good shape control and ensure that curves pass through their start and end points while providing simple distance computation that can handle specific requirements.

A. Evaluation setup

We have implemented our framework to determine the deployment of a swarm of 4 UAVs in a 3D space. The deployment space $\mathbf{\Omega}$ is assumed to be bounded by $[-4, 4]$ for the X and Y coordinates and $[0, 4]$ for the Z coordinate. The UAVs' velocity is limited to 2. They are deployed to monitor the events placed in the X-Y plane of the space $\mathbf{\Omega}$ using a continuous differentiable bi-variate normal distribution function f. Their initial positions are set randomly inside the deployment space boundaries. They must avoid colliding each other by maintaining a safe distance of 0.15. Additionally, they must establish and maintain a fully connected network, i.e., any two UAVs remain within a communication range of 2.2. The UAVs' target positions $\check{\boldsymbol{p}}_n(t)$, originally defined in (7), are determined through gradient ascent using

$$\check{\boldsymbol{p}}_n(t) = \boldsymbol{p}_n(t) + \gamma_n \nabla \boldsymbol{p}_n(t) C(\boldsymbol{p}_n(t)), \qquad (25)$$

where $C(\boldsymbol{p}_n(t))$ is defined as

$$C(\boldsymbol{p}_n(t)) = \sum_{n=1}^{N} \int_{\mathbf{\Omega}_n} f(\boldsymbol{\omega}) d\boldsymbol{\omega}. \qquad (26)$$

In this formulation, $\mathbf{\Omega}_n \subset \mathbf{\Omega}$ represents the disk-shaped projected area of UAV n on the X-Y plane and an indication of its covered area. This non-overlapping area is centered at the projected location of the UAV on the X-Y plane, with a radius of 0.36 times the UAV's height.

B. Results

We have evaluated our deployment framework using the setup in Sec. V-A. UAVs are deployed without collision and achieved a fully connected network. Their target positions $\check{\boldsymbol{p}}_n(t)$ maximized the performance function in (26) where the initial value and the final value are 2% and 80%, respectively. Fig. 4 shows the evolution of the performance function $C(\boldsymbol{p}_n(t))$.

Fig. 5 illustrates the evolution of distances between UAV pairs. As shown in the figure, the minimum distance constraint (collision) has consistently been satisfied throughout
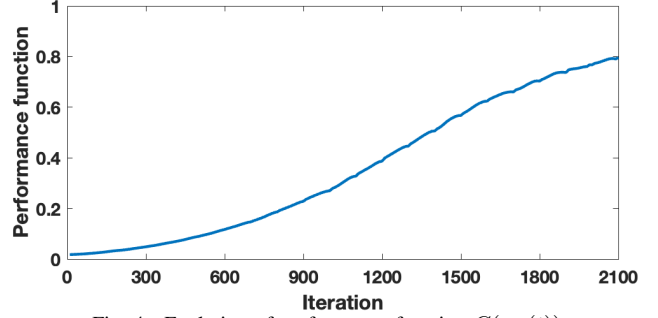
Fig. 4. Evolution of performance function $C(\boldsymbol{p}_n(t))$.

the deployment. The dashed blue line represents the threshold minimum distance of 0.15, which has never been breached by any two UAVs. The dashed orange line represents the threshold maximum distance of 2.0, which has been enforced on UAV pairs. Although the initial UAV network lacked full connectivity, for example the initial distance between UAVs 1 and 4 is more than 2.0, our framework effectively enforced the maximum distance constraint, resulting in the eventual creation of a fully connected network.

To demonstrate the effectiveness of our framework in enforcing the derivative related constraints, Fig. 6 shows the velocity of each UAV. The dashed green line represents the velocity threshold, set at 2, which has never been breached by any UAV.
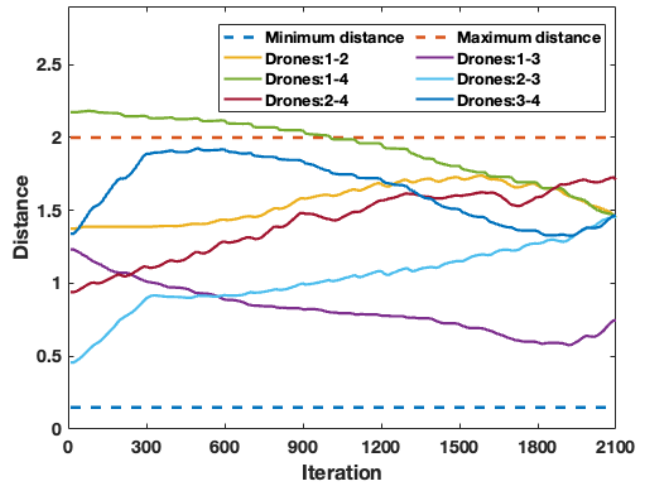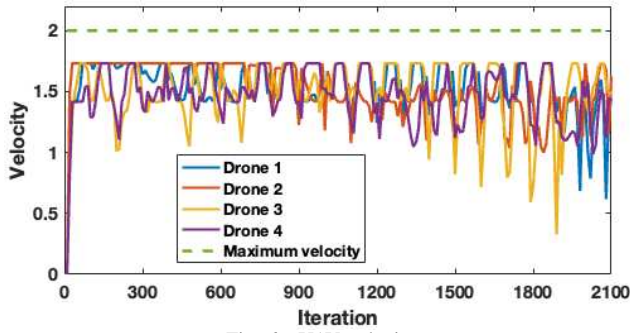
Fig. 5. Distance between pairs of drones.

Fig. 6. UAV velocity.

VI. CONCLUSIONS

In conclusion, this paper presents a framework for optimizing the deployment for a swarm of robots. This framework seamlessly transitions form 2D to 3D deployment, adopts the shift from design-time to real-time decision making, and accommodates complex robot/UAV dynamics. The trajectories are represented as Bézier curves, which are sampled as inputs to the robot/UAV dynamics to predict their states over a time horizon. Embedded within the geometry of Bézier curves are formulations for handling various constraints. This framework formulates the robot/UAV deployment as a quadratic programming problem, enforcing constraints and optimizing the deployment. Simulation results demonstrate promising performance of the proposed framework, offering a promising solution for deployment optimization.

Future research will focus on applying this platform to realistic use cases and testing it on the NSF-funded Aerial Experimentation and Research Platform for Advanced Wireless (AERPAW). Additionally, we will integrate this framework with various search algorithms to study and enhance the convergence of the controller. Another aspect of our future work is to incorporate various sensing and accuracy models to develop a robust deployment mechanism.

ACKNOWLEDGMENT

REFERENCES

[1] B. Zhu, E. Bedeer, H. H. Nguyen, R. Barton, and J. Henry, "UAV trajectory planning in wireless sensor networks for energy consumption minimization by deep reinforcement learning," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 9, pp. 9540–9554, 2021.

[2] S. Karimi-Bidhendi, J. Guo, and H. Jafarkhani, "Energy-efficient deployment in static and mobile heterogeneous multi-hop wireless sensor networks," *IEEE Transactions on Wireless Communications*, vol. 21, no. 7, pp. 4973–4988, 2022.

[3] T. Setter and M. Egerstedt, "Energy-constrained coordination of multi-robot teams," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 4, pp. 1257–1263, 2016.

[4] J. Guo, P. Walk, and H. Jafarkhani, "Optimal deployments of UAVs with directional antennas for a power-efficient coverage," *IEEE Transactions on Communications*, vol. 68, no. 8, pp. 5159–5174, 2020.

[5] C. Diaz-Vilor, A. Lozano, and H. Jafarkhani, "Cell-free UAV networks: Asymptotic analysis and deployment optimization," *IEEE Transactions on Wireless Communications*, vol. 22, no. 5, pp. 3055–3070, 2023.

[6] C. Diaz-Vilor, M. A. Almasi, A. M. Abdelhady, A. Celik, A. M. Eltawil, and H. Jafarkhani, "Sensing and communication in UAV cellular networks: Design and optimization," *IEEE Transactions on Wireless Communications*, vol. 23, no. 6, pp. 5456–5472, 2024.

[7] C. Diaz-Vilor, A. Lozano, and H. Jafarkhani, "Cell-free UAV networks with wireless fronthaul: Analysis and optimization," *IEEE Transactions on Wireless Communications*, vol. 23, no. 3, pp. 2054–2069, 2024.

[8] J. Guo and H. Jafarkhani, "Sensor deployment with limited communication range in homogeneous and heterogeneous wireless sensor networks," *IEEE Transactions on Wireless Communications*, vol. 15, no. 10, pp. 6771–6784, 2016.

[9] J. Guo and H. Jafarkhani, "Movement-efficient sensor deployment in wireless sensor networks with limited communication range," *IEEE Transactions on Wireless Communications*, vol. 18, no. 7, pp. 3469–3484, 2019.

[10] C. E. Luis, M. Vukosavljev, and A. P. Schoellig, "Online trajectory generation with distributed model predictive control for multi-robot motion planning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 604–611, 2020.

[11] L. Wang, A. D. Ames, and M. Egerstedt, "Safety barrier certificates for collisions-free multirobot systems," *IEEE Transactions on Robotics*, vol. 33, no. 3, pp. 661–674, 2017.

[12] T. Baca, D. Hert, G. Loianno, M. Saska, and V. Kumar, "Model predictive trajectory tracking and collision avoidance for reliable outdoor deployment of unmanned aerial vehicles," in *International Conference on Intelligent Robots and Systems*, pp. 6753–6760, IEEE, 2018.

[13] J. S. Kumar, S. K. Pandey, M. A. Zaveri, and M. Choksi, "Geo-fencing technique in unmanned aerial vehicles for post disaster management in the internet of things," in *International Conference on Advanced Computational and Communication Paradigms (ICACCP)*, pp. 1–6, IEEE, 2019.

[14] M. Medvedev, V. Pshikhopov, B. Gurenko, and N. Hamdan, "Path planning method for mobile robot with maneuver restrictions," in *International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)*, pp. 1–7, IEEE, 2021.

[15] D. J. Braun, F. Petit, F. Huber, S. Haddadin, P. Van Der Smagt, A. Albu-Schäffer, and S. Vijayakumar, "Robots driven by compliant actuators: Optimal control under actuation constraints," *IEEE Transactions on Robotics*, vol. 29, no. 5, pp. 1085–1101, 2013.

[16] H. F. Parapari, F. Abdollahi, and M. B. Menhaj, "Distributed coverage control for mobile robots with limited-range sector sensors," in *International Conference on Advanced Intelligent Mechatronics (AIM)*, pp. 1079–1084, IEEE, 2016.

[17] A. Saeed, A. Abdelkader, M. Khan, A. Neishaboori, K. A. Harras, and A. Mohamed, "Argus: realistic target coverage by drones," in *Proceedings of the International Conference on Information Processing in Sensor Networks*, pp. 155–166, 2017.

[18] Y. Morsly, N. Aouf, M. S. Djouadi, and M. Richardson, "Particle swarm optimization inspired probability algorithm for optimal camera network placement," *IEEE Sensors Journal*, vol. 12, no. 5, pp. 1402–1412, 2011.

[19] P. E. Bézier, "How Renault uses numerical control for car body design and tooling," tech. rep., SAE Technical Paper, 1968.

[20] H. Oruç and G. M. Phillips, "q-bernstein polynomials and Bézier curves," *Journal of Computational and Applied Mathematics*, vol. 151, no. 1, pp. 1–12, 2003.

[21] K. I. Joy, "Bernstein polynomials," *On-Line Geometric Modeling Notes*, vol. 13, no. 5, 2000.

[22] B.-G. Lee and Y. Park, "Distance for Bézier curves and degree reduction," *Bulletin of the Australian Mathematical Society*, vol. 56, no. 3, pp. 507–515, 1997.

[23] J. Zhou, E. C. Sherbrooke, and N. M. Patrikalakis, "Computation of stationary points of distance functions," *Engineering with computers*, vol. 9, pp. 231–246, 1993.

[24] E. Koyuncu and H. Jafarkhani, "Asynchronous local construction of bounded-degree network topologies using only neighborhood information," *IEEE Transactions on Communications*, vol. 67, no. 3, pp. 2101–2113, 2018.

[25] G. Farin, *Curves and surfaces for CAGD: a practical guide*. Elsevier, 2001.