Fast and In Sync: Periodic Swarm Patterns for Quadrotors

Xintong Du, Carlos E. Luis, Marijan Vukosavljev, and Angela P. Schoellig

***Abstract*— This paper aims to design quadrotor swarm performances, where the swarm acts as an integrated, coordinated unit embodying moving and deforming objects. We divide the task of creating a choreography into three basic steps: designing swarm motion primitives, transitioning between those movements, and synchronizing the motion of the drones. The result is a flexible framework for designing choreographies comprised of a wide variety of motions. The motion primitives can be intuitively designed using few parameters, providing a rich library for choreography design. Moreover, we combine and adapt existing goal assignment and trajectory generation algorithms to maximize the smoothness of the transitions between motion primitives. Finally, we propose a correction algorithm to compensate for motion delays and synchronize the motion of the drones to a desired periodic motion pattern. The proposed methodology was validated experimentally by generating and executing choreographies on a swarm of 25 quadrotors.**

I. INTRODUCTION

In recent years, robots have gradually found their way into the entertainment industry. Due to the advancements in both software and hardware, autonomous quadrotors, for example, have become active participants in art work and entertainment activities, showing off their exceptional agility in navigating in the three-dimensional space. Companies such as *Intel* and *SKYMAGIC* have delivered dazzling drone shows, where hundreds or even thousands of drones equipped with dedicated LEDs were coordinated to display enormous animations in the sky. Other companies, such as *Verity Studios* and *ElevenPlay*, have taken their shows to a different level, involving coordinated movements of drones with human performers, light effects, and music.

Drone shows where visual effects dominate the audience's experience have been pushed to an unprecedented scale and flown as many as 2018 drones [1]. However, shows that primarily rely on the effect of highly dynamic motions and fully leverage the motion capabilities of quadrotors remain at a much smaller scale. One reason is that drone swarms performing highly dynamical motions demand fast and accurate state estimates, motion planners and controllers, as well as an efficient and reliable overall flight control system architecture (e.g. with little or predictable time delays). Moreover, from a producer's perspective, the design and choreography of attractive, highly dynamic, swarm-like motion patterns for a large troupe of drones becomes more challenging due to inter-agent collision constraints and spatial constraints, as well as aerodynamic effects.

Authors are with the Dynamic Systems Lab (www.dynsyslab.org) at the University of Toronto Institute for Aerospace Studies (UTIAS), Canada. Email: xintong.du@mail.utoronto.ca, carlos.luis@mail.utoronto.ca, mario.vukosavljev@mail.utoronto.ca, schoellig@utias.utoronto.ca

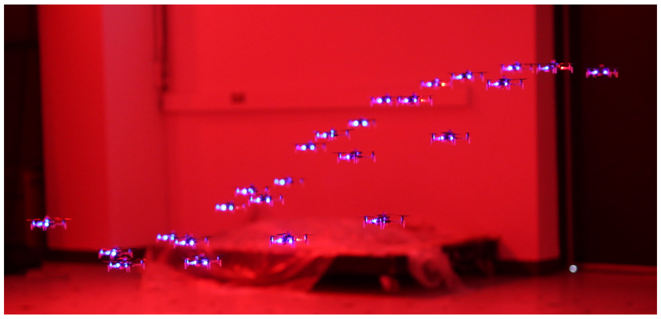

Fig. 1. Twenty-five quadrotors perform a periodic wave motion in the vertical direction. A video of a full performance is available at `http://tiny.cc/fast-periodic`.

Therefore, our work aims to provide motion planning and control strategies that enable a quadrotor swarm to act as an integrated entity and display highly dynamic motion patterns (see Fig. 1).

In the context of motion planning for complex task specifications, motion primitives have proven to be a successful framework. Motivated by creating theatrical performances, we aim to design a library of motion primitives, from which choreographers can then choose and combine motion primitives to form a coherent story. First, individual motion primitives must be designed. Rhythmic motion primitives for individual vehicles were proposed in [2]. A set of representative behaviour descriptors in [3] was designed to enable online interaction with a small group of robots. However, a unified framework for encoding a library of motion primitives for swarms is lacking. Next, to concatenate swarm motion primitives, we develop an algorithm for smooth and safe transitions. Many techniques in the robot planning and control literature already exist to transition vehicles from one configuration to another. Of particular interest is a simultaneous goal assignment and trajectory planning algorithm with a simple collision avoidance scheme proposed in [4], which has been applied in the context of drone performances [5]. Despite its scalability to large swarms and its consideration of vehicle dynamics at an early stage in the planning process, this method does not immediately apply to our situation. Namely, it does not guarantee a common start and end time for all vehicles, and it assumes the vehicles are at rest at the start and end, which makes designing smooth trajectories difficult for our dynamic motions. Finally, once motion primitives and their transitions are designed, they need to be executed reliably by the real system. To this end, a phase comparator along with a correction algorithm were proposed in [6] to eliminate the phase error in tracking the desired position trajectories.

The contributions of this paper are as follows. First, we present the generic formulation for swarm motion primitives that is suitable for describing periodic, highly dynamic motion patterns, where drones appear as a coordinated unit, for example, embodying moving and deforming objects. Second, we adapt and combine various state-of-the-art trajectory planning methods to safely and smoothly connect arbitrary swarm motion primitives and, additionally, preserve patterns in the swarm behaviour that could emerge from strategic goal assignment. Third, we show that a simple correction algorithm is capable of compensating for amplitude and phase errors that arise when trying to synchronize a swarm's motion to a desired periodic motion pattern. Finally, results are demonstrated experimentally on a large swarm of drones.

II. DYNAMIC, PERIODIC MOTION PATTERNS

This section presents our design of dynamic, periodic motion primitives for drone swarms. These motion primitives embody moving and deforming objects and are inspired by natural particle phenomena such as wave motions or rigid body rotations. The goal is that vehicles appear as an integrated entity.

A. Generic Formulation

We define a motion primitive for a swarm of N drones as a tuple

$$\mathcal{MP}^N = \left(t_0, t_f, \{\mathbf{r}^n\}_{n=1}^N, \mathcal{T}_d(\mathbf{r},t)\right), \tag{1}$$

which consists of the start time $t_0 \in \mathbb{R}$ and end time $t_f \in \mathbb{R}$ of the motion primitive, a unique characteristic configuration vector $\mathbf{r}^n \in \mathbb{R}^3$ for each drone $n = 1, \dots, N$, and a position trajectory generator $\mathcal{T}_d(\mathbf{r},t) : \mathbb{R}^3 \times [t_0, t_f] \to \mathbb{R}^3$. Then, the desired position evolution for the n-th drone over the interval $t \in [t_0, t_f]$ is given as $\mathbf{x}_d^n(t) = \mathcal{T}_d(\mathbf{r}^n, t)$.

As in [2], we parameterize the trajectory generator $\mathcal{T}_d$ as a Fourier series in order to demonstrate rhythmic motions. Moreover, we use the configuration vectors $\mathbf{r}^n$ to emphasize how each drone's motion is related to the overall display.

In this work, we define the trajectory function as

$$\mathcal{T}_d(\mathbf{r},t) = \mathcal{C}(\mathbf{r}) + \sum_{m=1}^{M} \left(\mathcal{A}_m(\mathbf{r})\sin(\omega_m t) + \mathcal{B}_m(\mathbf{r})\cos(\omega_m t)\right), \tag{2}$$

with frequencies $\omega_m \in \mathbb{R}$ as well as parameter functions $\mathcal{C}$, $\mathcal{A}_m$ and $\mathcal{B}_m : \mathbb{R}^3 \to \mathbb{R}^3$ that define the centre and amplitude of the sinusoidal functions in 3D. Note that m indexes motions with different frequencies and that we only consider a superposition of M periodic motions.

The parameters in (2) reflect desired rhythmic and visual effects. The temporal components ω_m describe intervals of repeated patterns. The spatial components $\mathcal{C}$, $\mathcal{A}_m$, $\mathcal{B}_m$, and $\{\mathbf{r}^n\}$ are responsible for the spatial pattern collectively demonstrated by the swarm. In particular, the parameter functions determine the overall picture presented to the audience, where each drone's contribution is reflected by their unique characteristic configuration vector.

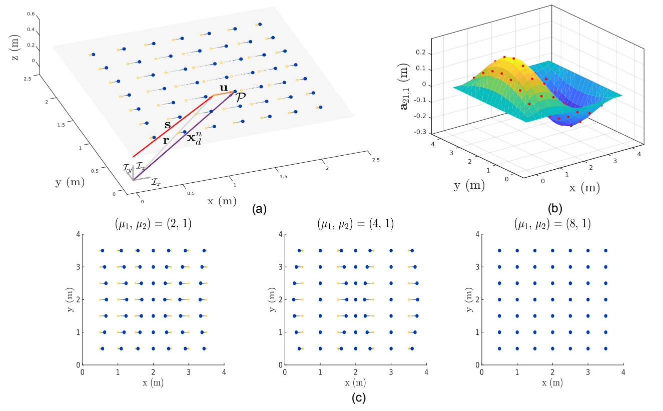

Fig. 2. This figure shows an example of a wave pattern on a bounded elastic surface represented by 49 drones positioned in a grid configuration. In (a), we show a schematic of a 1D wave motion over a quarter of a period where hollow circles and solid circles indicate the equilibrium and peak position, respectively. The color of the trajectories shifts from yellow to blue indicating the passage of time. In (b), the amplitude function $\mathcal{A}_m(\mathbf{r})$ in the x direction is $a_{\mu_1\mu_2,1}\sin(2\pi r_1)\sin(\pi r_2)$. Red dots are the amplitude evaluated at each characteristic vector $\mathbf{r}^n$. The wave motion with this amplitude function is demonstrated in the bottom left diagram in this figure. In (c), we illustrate the role of the index μ_1 (while keeping μ_2 fixed) in determining the wave motion's periodicity and symmetry.

Some possible choices for spatial patterns are given as below. In our experiments, we demonstrate two specific representations of (2), where vehicles represent the particle movement seen in waves and rigid body rotations.

B. Wave Patterns

Wave motions are ubiquitous in nature. Typical examples include surface water waves, sound traveling through air, and electromagnetic field propagation. They can be modeled by the wave propagation equation [7]. Consider a two-dimensional rectangular elastic surface with bounded edges as shown in Fig. 2a. Suppose the equilibrium location of each point $\mathcal{P}$ on the surface is parameterized by $\mathbf{s} = (s_1, s_2) \in [0,a] \times [0,b] \subset \mathbb{R}^2$. The disturbance to a point at time t, $\mathbf{u}(\mathbf{s},t) \in \mathbb{R}^3$, by a three-dimensional wave propagating through the surface at speed $c > 0$ is governed by

$$c^2\nabla^2\mathbf{u}(\mathbf{s},t) = \frac{\partial^2\mathbf{u}(\mathbf{s},t)}{\partial t^2}, \tag{3}$$

where ∇^2 is the spatial Laplacian. Its solution can be expressed as a product of spatial and temporal components:

$$\mathbf{u} = \sum_{(\mu_1,\mu_2)} \mathbf{a}_{\mu_1\mu_2}\sin\left(\frac{\mu_1}{a}\pi s_1\right)\sin\left(\frac{\mu_2}{b}\pi s_2\right)\sin(\omega_{\mu_1\mu_2}t) + \mathbf{b}_{\mu_1\mu_2}\sin\left(\frac{\mu_1}{a}\pi s_1\right)\sin\left(\frac{\mu_2}{b}\pi s_2\right)\cos(\omega_{\mu_1\mu_2}t), \tag{4}$$

where the amplitudes $\mathbf{a}_{\mu_1\mu_2}, \mathbf{b}_{\mu_1\mu_2} \in \mathbb{R}^3$ can be determined from the surface's initial position and velocity and frequencies $\omega_{\mu_1\mu_2} \in \mathbb{R}$ for $\mu_1, \mu_2 = 1, 2, 3...$ are dictated by the dispersion relation $\omega^2_{\mu_1\mu_2} = c^2\pi^2(\frac{\mu_1^2}{a^2} + \frac{\mu_2^2}{b^2})$.

Finally, we can use a finite approximation to this solution to generate reference trajectories for N drones situated on

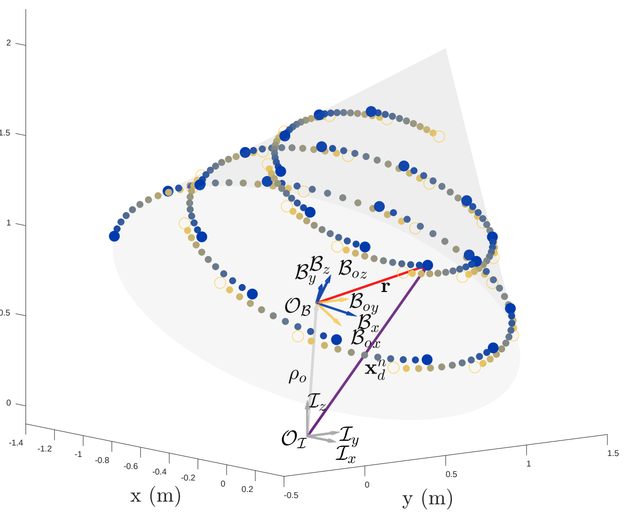

Fig. 3. An example of a rigid body rotation of which the characteristic vectors $\{\mathbf{r}^n\}_{n=1}^N$ are designed to form a helix on the surface of a tilted cone. Desired trajectories of each drone $\mathcal{T}_d(\mathbf{r}, t)$ over a time interval $t \in [t_1, t_2]$ are shown, where hollow circles and solid circles mark the start and end position, $\mathcal{T}_d(\mathbf{r}, t_1)$, $\mathcal{T}_d(\mathbf{r}, t_2)$, respectively. Color changes from yellow to blue along each trajectory to indicate the passage of time.

the surface. Suppose that we select M pairs of (μ_1, μ_2) terms in the finite approximation and assign a unique point $\mathbf{r}^n \in [0, a] \times [0, b] \times \{h\}$ to each drone $n = 1, \ldots, N$ on the surface at some desired height $h > 0$. Then the desired reference is $\mathbf{x}_d^n(t) = \mathbf{r}^n + \mathbf{u}((r_1^n, r_2^n), t)$ for $n = 1, \ldots, N$. Comparing to (1) and (2), we can design a swarm motion primitive, where the indices $m = 1, \ldots, M$ correspond to a pair (μ_1, μ_2), the frequencies are $\omega_m = \omega_{\mu_1\mu_2}$, and parameters are $\mathcal{C}(\mathbf{r}) = \mathbf{r}$, $\mathcal{A}_m(\mathbf{r}) = \mathbf{a}_{\mu_1\mu_2} \sin\left(\frac{\mu_1}{a}\pi r_1\right) \sin\left(\frac{\mu_2}{b}\pi r_2\right)$ and $\mathcal{B}_m(\mathbf{r}) = \mathbf{b}_{\mu_1\mu_2} \sin\left(\frac{\mu_1}{a}\pi r_1\right) \sin\left(\frac{\mu_2}{b}\pi r_2\right)$.

The surface wave (4) has desirable geometric properties, such as symmetry and periodicity in its spatial component $\mathcal{A}_m$, $\mathcal{B}_m$. However, the parameters $\mathbf{r}^n$, μ_1, μ_2, $\mathbf{a}_m$ and $\mathbf{b}_m$ must be carefully selected when representing a pattern on the continuous rectangular surface with a finite number of drones. To be specific, μ_1 and μ_2 determine the spatial frequency of the oscillation amplitude and, thus, the symmetry and periodicity in the overall wave pattern. The pattern gets more interesting as the axes or points of symmetry increase until their spacing is smaller than that between drones. The amplitudes $\mathbf{a}_m$, $\mathbf{b}_m$ do not necessarily have three non-zero components: in fact, the pattern may be perceived as more visually appealing if, for instance, motion in the horizontal plane is decoupled from vertical motion. Although any distribution of the drones within the surface is valid and should not influence the overall pattern, a selection of the vectors $\{\mathbf{r}^n\}_{n=1}^N$ that shares similar symmetry as $\mathcal{A}_m$, $\mathcal{B}_m$ may offer a better visual experience to the audience, such as a radial or rectangular mesh. Examples illustrating the role of each parameter are shown in Fig. 2b and Fig. 2c.

C. Rigid Body Rotation

A rigid body rotation can also be expressed in the form (2). Consider a rigid body $\mathcal{V}$ rotating in an inertial frame $\mathcal{F}_\mathcal{I}$ at a constant angular velocity, for example, the tilted cone in Fig. 3. First we attach a body frame $\mathcal{F}_\mathcal{B}$ with origin $\boldsymbol{\rho}_o$ relative to the inertial frame $\mathcal{F}_\mathcal{I}$. Next, we define an inertially fixed frame $\mathcal{F}_{\mathcal{B}_o}$ which overlaps with $\mathcal{F}_\mathcal{B}$ at time $t_o = 0$.

Let $\mathbf{r} \in \mathbb{R}^3$ be the position of a point expressed in $\mathcal{F}_\mathcal{B}$. Then this point is expressed in the inertial frame $\mathcal{F}_\mathcal{I}$ as

$$\mathbf{r}^\mathcal{I} = \boldsymbol{\rho}_o + \mathcal{R}_{\mathcal{I}\mathcal{B}_o}\mathcal{R}_{\mathcal{B}_o\mathcal{B}}\,\mathbf{r}, \tag{5}$$

where $\mathcal{R}_{\mathcal{I}\mathcal{B}_o} = [\mathbf{e}_1\,\mathbf{e}_2\,\mathbf{e}_3]$ is the rotation matrix from $\mathcal{F}_{\mathcal{B}_o}$ to $\mathcal{F}_\mathcal{I}$ and, without loss of generality, $\mathcal{R}_{\mathcal{B}_o\mathcal{B}}$ is a principle rotation along the z axis of $\mathcal{F}_{\mathcal{B}_o}$ given by $\mathcal{R}_{\mathcal{B}_o\mathcal{B}} = \mathcal{R}_Z(\omega_z t)$. Expanding (5), we obtain the trajectory generator of a rigid body rotation:

$$\begin{aligned}\mathcal{T}_d(\mathbf{r}, t) = \boldsymbol{\rho}_o + \mathbf{e}_3 r_3 + &(\mathbf{e}_2 r_1 - \mathbf{e}_1 r_2)\sin(\omega_z t) + \\ &(\mathbf{e}_1 r_1 + \mathbf{e}_2 r_2)\cos(\omega_z t).\end{aligned} \tag{6}$$

This may now be compared with (2) to obtain the parameters and frequencies.

Since ω_z determines the periodicity and $\boldsymbol{\rho}_o$ is simply the translation of the rotating object, the characteristic vectors $\{\mathbf{r}^n\}_{n=1}^N$, which represent the drones' distribution in the rotating body, are the only parameters to be designed. Due to a limited number of drones as well as the constraints on the minimum distance among them, $\{\mathbf{r}^n\}_{n=1}^N$ should be strategically chosen in order to have the audience recognize the shape of the body. For instance, it is easier to identify a cone if the drones outline a helix lying on the surface of the cone, as opposed to spacing them uniformly.

III. TRANSITION TRAJECTORY PLANNING

This section presents our transition trajectory planner that coordinates the swarm to make smooth and safe transitions.

A. Problem Statement

Consider two consecutive motion primitives $\mathcal{MP}_1^N$ and $\mathcal{MP}_2^N$ defined as

$$\begin{aligned}\mathcal{MP}_1^N &= \left(t_{0,1}, t_{f,1}, \{\mathbf{r}_1^n\}_{n=1}^N, \mathcal{T}_{d,1}(\mathbf{r}, t)\right)\\ \mathcal{MP}_2^N &= \left(t_{0,2}, t_{f,2}, \{\mathbf{r}_2^n\}_{n=1}^N, \mathcal{T}_{d,2}(\mathbf{r}, t)\right).\end{aligned}$$

In what follows, we assume a given common start time t_s and end time t_e for all drone actors, where $|t_e - t_{0,2}| \leqslant \epsilon$ for a small number ϵ. We aim to compute:

- An assignment $\mathcal{M} : \{\mathbf{r}_1^n\}_{n=1}^N \to \{\mathbf{r}_2^n\}_{n=1}^N$ that assigns each drone identified by $\mathbf{r}_1^n$ in $\mathcal{MP}_1$ a unique characteristic configuration vector $\mathbf{r}_2^n$ in $\mathcal{MP}_2$.
- For each drone identified as $\mathbf{r}_1^n$ in $\mathcal{MP}_1$, a smooth trajectory $\mathcal{T}_s^n(t)$ from $\mathcal{MP}_1$ to $\mathcal{MP}_2$ that respects the quadrotors' motion constraints, the flight space boundary and inter-agent collision constraints.

We address the trajectory planning problem in two steps: *(i)* goal assignment and *(ii)* trajectory generation. As in [4], [5], we employ a collision resolution strategy in the second step, but we additionally enforce a common start and arrival time, as well as the smoothness of the resulting collision-free trajectories to achieve smooth and coordinated transitions.

B. Goal Assignment

We formulated the goal assignment problem as a combinatorial optimization problem. In our application, the assignment $\mathcal{M}$ aims to maximize the smoothness of trajectories generated in the next step. More complicated swarm transition objectives can be defined, such as minimizing the likelihood of collisions. However, in this case, the cost function of one assignment depends on how other assignments are made, which results in a challenging nonlinear optimization problem.

If we denote the cost of assigning $\mathbf{r}_1^\alpha$ to $\mathbf{r}_2^\beta$ as $\mathcal{J}_a(\mathbf{r}_1^\alpha, \mathbf{r}_2^\beta)$, the mapping $\mathcal{M}$ can be found by solving the linear assignment problem

$$\mathcal{M} = \underset{\mathcal{M}(\cdot)}{\operatorname{argmin}} \sum_{n=1}^{N} \mathcal{J}_a(\mathbf{r}_1^n, \mathcal{M}(\mathbf{r}_1^n)) \tag{7}$$

using the Hungarian algorithm [8]. To obtain the assignment cost $\mathcal{J}_a(\mathbf{r}_1^n, \mathcal{M}(\mathbf{r}_1^n))$, we solve a simplified minimum snap trajectory generation problem as in [9], for which only state continuity constraints are enforced. We then take the optimal cost function value of that problem as the assignment cost, namely,

$$\begin{aligned} \mathcal{J}_a(\mathbf{r}_1^\alpha, \mathbf{r}_2^\beta) = \min_{\mathcal{T}_{\alpha,\beta}(\cdot)} \quad & \int_{t_s}^{t_e} (\mathcal{T}_{\alpha,\beta}{}^{(4)}(\tau))^2 d\tau \\ \text{s.t.} \quad & \mathcal{T}_{\alpha,\beta}{}^{(p)}(t_s) = \mathcal{T}_{d,1}{}^{(p)}(\mathbf{r}_1^\alpha, t_s), \\ & \mathcal{T}_{\alpha,\beta}{}^{(p)}(t_e) = \mathcal{T}_{d,2}{}^{(p)}(\mathbf{r}_2^\beta, t_e), \end{aligned} \tag{8}$$

with $p = 0, 1, 2, 3, 4$. Note that $\mathcal{T}_{\alpha,\beta}(t) : [t_s, t_e] \to \mathbb{R}^3$ is the transition trajectory assigning $\mathbf{r}_1^\alpha$ to $\mathbf{r}_2^\beta$. We parameterize it using a single polynomial curve of P^{th} order in each direction

$$\mathcal{T}_{\alpha,\beta}(t) = \left[\sum_{p=0}^{P} a_{p,x} t^n \quad \sum_{p=0}^{P} a_{p,y} t^n \quad \sum_{p=0}^{P} a_{p,z} t^n\right]^T. \tag{9}$$

It can be shown that the minimum snap cost function can be written in quadratic form [10]. Therefore, (8) can be solved efficiently.

C. Collision-Free Smooth Trajectory Generation

Given $\mathcal{MP}_1^N$, $\mathcal{MP}_2^N$ and the assignment $\mathcal{M}$, we aim to find a smooth and collision-free trajectory $\mathcal{T}_s^n$ for each drone. Inspired by [11], we decouple the problem into N subproblems to avoid accounting for the collision constraints in a large joint space. However, our proposed algorithm consists of two steps: *(i)* find a dynamically feasible candidate trajectory for each drone denoted as $\mathcal{T}_c^n(t)$ and *(ii)* iteratively resolve collisions in $\{\mathcal{T}_c^n\}_{n=1}^N$ (if any) in a sequential manner.

1) Generating Candidate Trajectories: We solve the full single-agent minimum snap trajectory generation problem defined in [9] to generate candidate trajectories $\{\mathcal{T}_c^n(t)\}_{n=1}^N$ for the assignment $\mathcal{M}$. The candidate trajectory $\mathcal{T}_c^n(t)$ is parameterized in the same way as (9). Then, the optimization problem is given as

$$\begin{aligned} \min_{\mathbf{x}_c} \quad & \frac{1}{2} \mathbf{x}_c{}^{\mathbf{T}} \mathbf{H} \mathbf{x}_c \\ \text{s.t.} \quad & \mathcal{T}_c{}^{(p)}(t_s) = \mathcal{T}_{d,1}{}^{(p)}(\mathbf{r}_1^n, t_s), \\ & \mathcal{T}_c^{n(p)}(t_e) = \mathcal{T}_{d,2}{}^{(p)}(\mathcal{M}(\mathbf{r}_1^n), t_e), \\ & \mathbf{x}_{lb} \leqslant \mathcal{T}_c^n(t_k) \leqslant \mathbf{x}_{ub}, \\ & \mathbf{v}_{lb} \leqslant \mathcal{T}_c^{n(1)}(t_k) \leqslant \mathbf{v}_{ub}, \\ & T_{lb} \leqslant ||\mathcal{T}_c^{n(2)}(t_k)||_2 \leqslant T_{ub}, \\ & \mathbf{J}_{lb} \leqslant \mathcal{T}_c^{n(3)}(t_k) \leqslant \mathbf{J}_{ub}, \end{aligned} \tag{10}$$

with $p = 0, 1, 2, 3, 4$, $n = 1, 2, ..., N$ and $k = 1, 2, ..., K$. The decision vector $\mathbf{x}_c$ consists of concatenated polynomial coefficients of all drones $\mathbf{x}_c = [a_{0,x}^1 ... a_{P,x}^1\, a_{0,y}^1 ... a_{P,y}^1 \, ... \, a_{0,z}^N ... a_{P,z}^N]$. The matrix $\mathbf{H}$ is obtained by taking the Hessian of the minimum snap cost function defined in (8), while $\mathbf{x}$, $\mathbf{v}$, $\mathbf{J} \in \mathbb{R}^3$ and $T \in \mathbb{R}$, represent position, velocity, jerk and normalized thrust, respectively. The subscripts lb and ub refer to lower and upper bounds. We enforce the state constraints at discrete time steps $t_k = t_s + k(t_e - t_s)/K$.

2) Iterative Collision Resolution: Given the candidate trajectories $\{\mathcal{T}_c^n\}_{n=1}^N$, we construct a directed collision graph $G = \{V, E\}$. A vertex $v_n \in V$ represents the drone identified by $\mathbf{r}_1^n$, and an edge $e_{nm} \in E$, $n > m$ points from v_n to v_m, indicating a collision between $\mathcal{T}_c^n$ and $\mathcal{T}_c^m$. For any edge e_{nm}, we force the collision avoidance maneuver to be executed solely by drone n, while drone m follows its intended path. Our goal is to remove all edges $e_{nm} \in E$ by finding a collision-free trajectory $\mathcal{T}_s^n$ for drone n.

In particular, we adjust the candidate trajectory $\mathcal{T}_c^n$ by solving an optimization problem with a new cost function and additional ellipsoid collision constraints. Define $\mathcal{H}(\mathbf{x}_c)$ and $\mathcal{F}(\mathbf{x}_c)$ to be the set of $\widetilde{m}$ equality constraints and $\widetilde{p}$ inequality constraints in (10), respectively. Then, the collision avoidance problem for drone n is given as

$$\begin{aligned} \min_{\mathbf{x}_s^n} \quad & (\mathbf{x}_s^n - \mathbf{x}_c^{n*})^{\mathbf{T}} \mathbf{W} (\mathbf{x}_s^n - \mathbf{x}_c^{n*}) \\ \text{s.t.} \quad & \mathcal{H}_i(\mathbf{x}_s^n) = 0, \quad i = 0, 1, \ldots, \widetilde{m}, \\ & \mathcal{F}_i(\mathbf{x}_s^n) \leqslant 0, \quad i = 0, 1, \ldots, \widetilde{p}, \\ & 2 \leqslant ||\mathbf{E}^{-1}(\mathcal{T}_s^n(t_k) - \mathcal{T}_c^m(t_k))||_2^2, \end{aligned} \tag{11}$$

where $\mathcal{T}_s^n$ is parameterized as in (9) with coefficient vector $\mathbf{x}_s^n \in \mathbb{R}^{3(P+1)}$ as the decision variable of the optimization. The vector $\mathbf{x}_c^{n*}$ consists of the coefficients of $\mathcal{T}_c^n$ obtained in the first step, $\mathbf{W}$ is a weighting matrix trading off the deviation of $\mathcal{T}_s^n$ from $\mathcal{T}_c^n$ in three dimensions, $\mathbf{E} \in \mathbb{R}^3$ is a diagonal matrix specifying the radii of the ellipsoid boundary around the drones [12], and $m \in \{m \in \mathbb{N} \mid m \neq n, m \leqslant N\}$. Note that in (11) we enforce the same constraints as in (10) to guarantee the smoothness of trajectories after resolving the collisions.

We solve (11) sequentially for drones in V in decreasing order of the amount of outbound edges. If an optimal solution is found for v_n, we remove all of its outbound edges; otherwise, in cases where the problem is temporarily

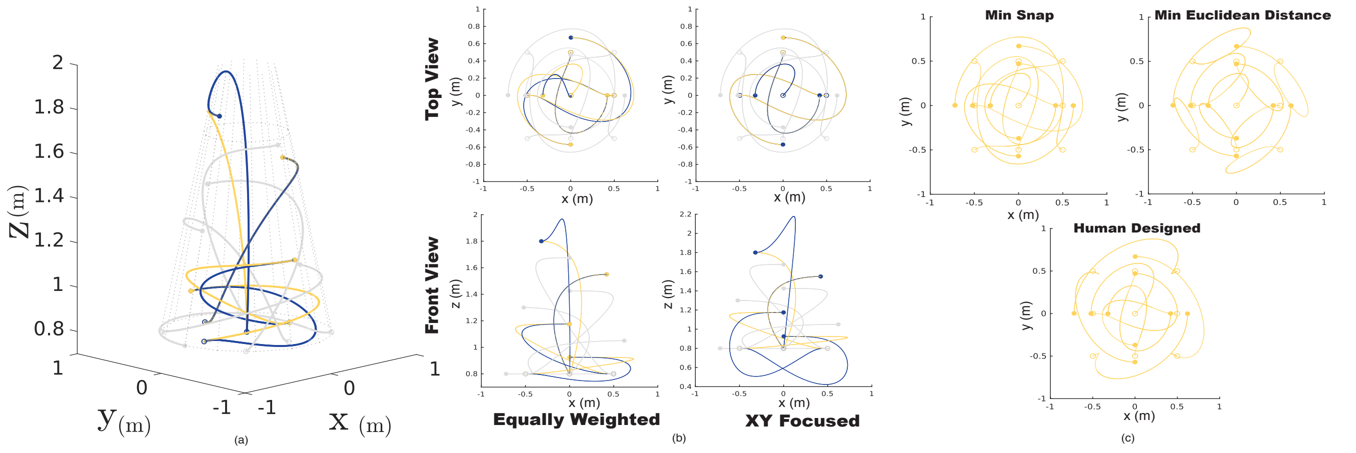

Fig. 4. An example of collision-free smooth trajectory generation for 9 drones from a wave pattern to a rigid body rotation similar to the ones in Fig. 2 and Fig. 3, respectively. Hollow circles and solid circles mark the start and end of the transition trajectories, $\mathcal{T}_{d,1}(t_s)$ and $\mathcal{T}_{d,2}(t_e)$. In (a), results of the trajectory generation involving four colliding candidate trajectories $\mathcal{T}_c$ (yellow) were adapted sequentially until we obtain the collision-free $\mathcal{T}_s$ (blue). Collision-free candidate trajectories are shown in grey. In (b), we illustrate the role of weighting factor $\mathbf{W}$ in (11) towards the objective of coordinated transitioning. Trajectories are colored in the same way as in (a). Finally, (c) shows comparisons between different goal assignment methods in facilitating the generation of trajectory candidates.

infeasible or hard to find a solution (e.g., other drones are not in favourable positions), we skip to the next drone. This procedure is repeated until either E is empty or a maximum number of iterations is exceeded. In the latter case, we turn to the coupled optimization approach in [13], which has shown a high probability of finding a solution.

D. Design Considerations

In order to generate smooth transition trajectories, a few key design decisions were made that differentiate our method from those in previous work. First, the vehicle dynamics and transition time are incorporated into the assignment cost function $\mathcal{J}_a$ in order to facilitate smooth trajectory generation in the following steps. Second, trajectories are parameterized with a single polynomial curve instead of concatenated polynomials to minimize unnecessary curvature. Finally, continuity in trajectories at t_s and t_e is guaranteed in the collision resolution step. We illustrate the first design feature in the top two figures of Fig. 4c, which highlights the difference between our assignment cost functions (minimum snap) and the Euclidean distance used in [3]. The former cost function induces smooth, fluid, and energy-saving candidate trajectories as compared to the latter.

The bottom figure in Fig. 4c shows a strategic manual assignment, where rotational symmetry of the initial and final motion primitives were incorporated. Consequently, it introduces nice geometric properties in the candidate trajectories that enable a coordinated transition process. However, incorporating these strategic assignments into our framework is non-trivial and would result in nonlinear optimization problems. Nonetheless, if we do have particular features in the candidate trajectories, we are able to preserve them during the collision resolution step by penalizing the difference between the final and candidate trajectories in (11). Moreover, we introduced the weighting matrix $\mathbf{W}$ to optionally preserve the swarm transition patterns in some dimensions while relaxing them in the others. An example is shown in Fig. 4b, where patterns in the $x-y$ plane are preserved during collision avoidance by encouraging motion in the z direction.

IV. Motion Synchronization

To execute the proposed highly-dynamic motion patterns in tight formations and in sync with the desired periodic motion pattern, a high-accuracy controller for periodic motions is required. In practice, one major issue that can be observed is the phase shift and amplitude amplification or attenuation [6] due to the delay in communicating vehicle commands, the inherent delay in dynamical systems, as well as the delay in sensor measurements. It is possible to reduce delay and amplitude error by adding feed-forward terms to the drones' position and attitude controller. However, it incurs communication overhead which is not ideal and sometimes impossible for a large swarm. Similarly to [6], we actively adjust the desired trajectories $\mathcal{T}_d(\mathbf{r},t)$ in (2) by scaling their amplitude and shifting them in time to obtain a reference trajectory.

If we approximate the quadrotor as a linear system in each spatial direction with transfer functions $H_i(s)$, $i=1,2,3$, we can estimate the amplitude attenuation and phase shift at each oscillating frequency ω_m from the system's frequency response to sinusoidal reference signals. The compensated reference trajectory in the i^{th} direction is given as

$$\mathcal{T}_{r,i}(\mathbf{r},t) = \mathcal{C}_i(\mathbf{r}) + \sum_{m=1}^{M} \kappa_{m,i}\mathcal{A}_{m,i}(\mathbf{r})\sin(\omega_m t + \phi_{m,i}) + \sum_{m=1}^{M} \kappa_{m,i}\mathcal{B}_{m,i}(\mathbf{r})\cos(\omega_m t + \phi_{m,i}), \tag{12}$$

where the amplitude scalings $\kappa_m \in \mathbb{R}^3$ and phase shifts

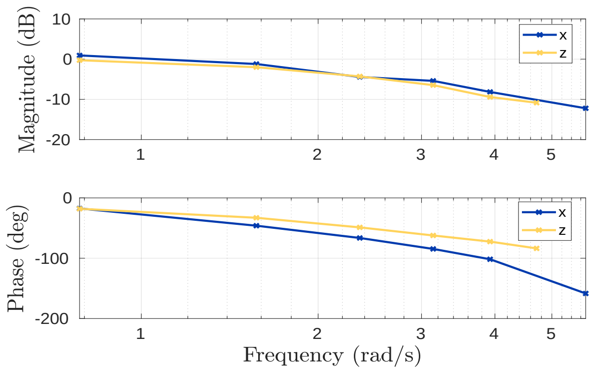

Fig. 5. The empirical Bode Plot for a Crazyflie 2.0 quadrotor.

$\phi_m \in \mathbb{R}^3$ can be determined from the closed-loop system's transfer function observed in experiments, $H_i(s)$. That is, we compute $\kappa_{m,i} = |H_i(j\omega_m)|^{-1}$, $\phi_{m,i} = -\arg H_i(j\omega_m)$.

V. EXPERIMENTAL RESULTS

We demonstrate our algorithms using the small-sized quadrotor platform Crazyflie 2.0 in a multi-agent testbed at the Dynamic Systems Lab (inspired by the Crazyswarm [14]). From a central computer, we gathered position data from all the drones in the fleet using an overhead motion capture system. The state information was sent via radio to each drone's onboard computer, along with the desired position, which is tracked using an onboard position controller.

A. Motion Synchronization

We empirically determined the Crazyflie's closed-loop transfer functions $H_i(js)$, $i = 1, 2, 3$, by commanding sinusoidal position trajectories at different frequencies ω in the x and z directions. Based on the phase shift and amplitude attenuation between the desired and actual trajectories, we found the system's frequency response characterized by amplitude attenuation $|H_i(j\omega)|$ and phase shift $\arg H_i(j\omega)$ as shown in Fig. 5. Based on the data, we constructed a look-up table from which the scaling factors for motion primitives, $\boldsymbol{\kappa}_m$ and $\boldsymbol{\phi}_m$, are determined by linearly interpolating points in the magnitude and phase mapping, respectively. We constructed the look-up table using data from one drone but evaluated it on another 8 drones. Fig. 6 summarizes the improvements on tracking performance of the 8 drones compared to when there is no synchronization.

B. Transition Trajectory Planning

The three optimization problems in (8), (10) and (11) are solved using the nonlinear optimizer IPOPT [15]. We used $P = 14$ as the polynomial order and $K = 10$ as a starting point for partition intervals, which may be doubled in the next iteration if a feasible solution is found but a collision occurs in between two time steps t_k and t_{k+1}. In practice, we found that the high-order polynomial trajectory when evaluated at, for example, $t_e = 7$, will introduce numerical problems. Therefore, a nondimensionalization technique as in [9] is necessary for efficiently solving the problem.

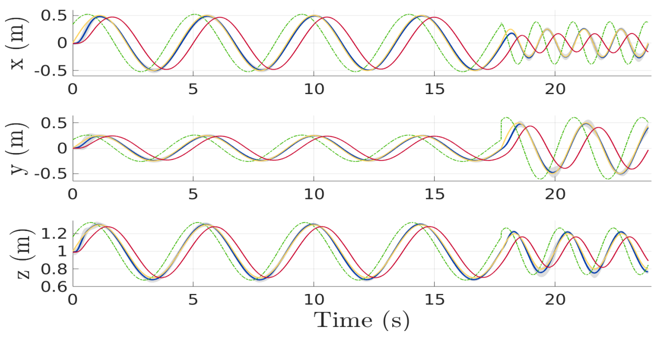

Fig. 6. Evaluation of the proposed motion synchronization algorithm on 8 drones. The desired trajectory (yellow) is adjusted in its amplitude and phase to obtain the reference trajectory (green). On average, the tracking response (blue) of the drones being tested is improved compared to the cases without compensation (red) with small variances shaded in grey.

We evaluated the transition trajectory planner in both simulation and experiments. In simulation, we tested with 25 drones transitioning in a $5 \times 5 \times 2$m volume with a collision radius of 0.14m in the $x - y$ plane and 0.35m in the z direction. Our algorithm found a feasible solution in 85% of 1800 randomly generated motion primitive pairs. The main reason for failure are numerical difficulties in solving (11) when the transition time is long. One repair strategy is to parameterize the trajectories with a few concatenated polynomials, instead of just one polynomial.

C. Choreographed Drone Performance

The video at `http://tiny.cc/fast-periodic` presents a one-minute drone performance where 25 drones are choreographed to fly in tight formations and perform fast and dynamic motions. We seamlessly concatenate five wave motions and one rigid body rotation to create a cohesive performance. The peak velocity and pitch angle reached 2.65m/s and 32 degrees, respectively. Since the motion primitives are aggressive, we used a larger collision ellipsoid than in [12] with 0.28m in $x - y$ and 0.85m in z.

VI. CONCLUSIONS

In this paper, we provide guidelines for creating performances with quadrotor swarms that fully leverage their motion capabilities to create appealing visual effects. The swarm motion primitives are formulated as coupled periodic motions, which are described by a single equation indicating the overall motion pattern and the relationship between the individual actors of the swarm. The geometric properties of parameter functions in this formulation were discussed. Moreover, we provided a hierarchical transition trajectory planner to seamlessly connect motion primitives together and preserve geometric characteristics. Lastly, a correction algorithm is proposed to improve the quadrotors' tracking performance of the periodic motions, which allows the swarm to perform synchronously and in close proximity to each other. The method is validated with a swarm performance of 25 drones in a compact space.

References

[1] Intel. (2018) Intel breaks guinness world records title for drone light shows in celebration of 50th anniversary. [Online]. Available: https://newsroom.intel.com/news/intel-breaks-guinness-world-records-title-drone-light-shows-celebration-50th-anniversary/.

[2] F. Augugliaro, A. P. Schoellig, and R. DAndrea, "Methods for designing and executing an aerial dance choreography," *IEEE Robot. Autom. Mag*, vol. 20, no. 4, pp. 96–104, 2013.

[3] E. A. Cappo, A. Desai, M. Collins, and N. Michael, "Online planning for human–multi-robot interactive theatrical performance," *Autonomous Robots*, pp. 1–16, 2018.

[4] M. Turpin, K. Mohta, N. Michael, and V. Kumar, "Goal assignment and trajectory planning for large teams of interchangeable robots," *Autonomous Robots*, vol. 37, no. 4, pp. 401–415, 2014.

[5] A. Desai, E. A. Cappo, and N. Michael, "Dynamically feasible and safe shape transitions for teams of aerial robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 5489–5494.

[6] A. Schöllig, F. Augugliaro, S. Lupashin, and R. D'Andrea, "Synchronizing the motion of a quadrocopter to music," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2010, pp. 3355–3360.

[7] H. Georgi, *The physics of waves*. Prentice Hall Englewood Cliffs, NJ, 1993.

[8] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[9] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2011, pp. 2520–2525.

[10] C. Richter, A. Bry, and N. Roy, "Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments," in *Robotics Research*. Springer, 2016, pp. 649–666.

[11] Y. Chen, M. Cutler, and J. P. How, "Decoupled multiagent path planning via incremental sequential convex programming," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 5954–5961.

[12] J. A. Preiss, W. Hönig, N. Ayanian, and G. S. Sukhatme, "Downwash-aware trajectory planning for large quadrotor teams," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 250–257.

[13] F. Augugliaro, A. P. Schoellig, and R. D'Andrea, "Generation of collision-free trajectories for a quadrocopter fleet: A sequential convex programming approach," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012, pp. 1917–1922.

[14] J. A. Preiss, W. Honig, G. S. Sukhatme, and N. Ayanian, "Crazyswarm: A large nano-quadcopter swarm," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 3299–3304.

[15] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, no. 1, pp. 25–57, 2006.