IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 3, 2019, accepted September 26, 2019, date of publication October 8, 2019,
date of current version December 11, 2019.

*Digital Object Identifier 10.1109/ACCESS.2019.2946273*

# Collision-Free Path Planning for Cooperative Aerial Manipulators Under Velocity and Curvature Constraints

**HYEONBEOM LEE[1], (Member, IEEE), CLARK YOUNGDONG SON[2], AND H. JIN KIM[2], (Member, IEEE)**

[1]School of Electronics Engineering, Kyungpook National University, Daegu 41566, South Korea
[2]School of Mechanical and Aerospace Engineering, Seoul National University, Seoul 08826, South Korea

Corresponding author: Hyeonbeom Lee (hbeomlee@knu.ac.kr)

This work was supported in part by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Korea Government (MSIT) under Grant NRF-2019R1F1A1062667, and in part by the Korea Institute of Machinery and Materials grant funded by the Korean Government under Grant NK217G.

**ABSTRACT** This paper presents a path planning problem with velocity and curvature constraints for cooperative aerial manipulators in obstacle environments. While potential-flow-based path planning approaches can help generate smooth paths in which cooperative aerial manipulators can avoid obstacles, they do not facilitate the consideration of velocity and curvature constraints. By contrast, while such constraints can be considered in the optimization approach, the approach often requires a heavy computational load. To overcome these drawbacks, we propose a simple approach based on curvature-constrained harmonic potential flow and a streamline-changing algorithm. We also handle the initial and final velocity problem by a smooth activation function, which is crucial for the tracking performance at the initial and final position. The proposed approach can be used to generate a smooth path for cooperative robots in real time. For the validation of the approach, the simulation results obtained with the proposed algorithm were compared with those acquired with an existing approach including dynamic movement primitives(DMPs). Furthermore, we performed load-carrying experiment using custom-made aerial manipulators.

**INDEX TERMS** Collision-free path planning, constraints, cooperative robots, multirotors, obstacle avoidance.

## I. INTRODUCTION

Cooperative manipulation enables handling of a heavy or bulky payload beyond the limits of a robot's transportation capabilities. Researchers have demonstrated various applications of ground mobile manipulators [1] and human-rocomputationally heavy optimizationbot collaboration [2]. Recently, owing to the affordability and simple hardware structure of multirotor, cooperative aerial manipulation has become an important research topics in the field of cooperative manipulation [3]–[11]. However, aerial cooperation involves complexity associated with multiple aerial robots such as coordination, synchronization of manipulators or stability due to the payload limits. For example, while avoiding obstacles during cooperative transportation, aerial robots might lose stability because of unknown disturbances such as interaction forces generated by other aerial manipulators.

This study is aimed at resolving the aforementioned coordination and planning algorithm for cooperative aerial manipulators in obstacle environments. Toward this end, a path planning algorithm is designed for cooperative aerial manipulators so that they can avoid obstacles and satisfy velocity and curvature constraints. To reduce the interaction force for the purpose of coordination, the specific distance between aerial robots should be maintained during flight. These algorithms are the main focus of this paper.

### A. CONTRIBUTION

The contributions of this paper can be summarized as follows. First, we propose a simple and efficient framework to consider velocity and curvature constraints for cooperative aerial manipulators. Unlike existing works for the constrained optimization approach for single [12] or multiple drones [6],

The associate editor coordinating the review of this manuscript and approving it for publication was Emre Koyuncu.

we develop a real-time motion generation algorithm for constrained environments without using a computationally-expensive optimization approach. A smooth path can be generated by exploiting smooth activati function and a harmonic potential field. To the best of the authors's knowledge, there is no comparable published work on velocity and curvature constrained path planning for cooperative aerial manipulators. We also perform a thorough analysis and a simulation to validate the final velocity and the stability concerning internal forces. In comparison with the algorithm presesnted in our previous work [11], the proposed algorithm generates a zero-speed path at the final destination by using a smooth activation function and can reduce internal forces while enabling the cooperative aerial manipulators to avoid obstacles. Second, we implement the proposed algorithm on custom-made aerial manipulators. Experimental result shows the feasibility of the proposed algorithm in enabling cooperative aerial manipulators to jointly carry a payload.

### B. RELATED RESEARCH

The use of cooperative aerial manipulation dates back to early works using a towed-cable [4]. In [4], the authors used multiple aerial robots for transportation via cables, and they considered the tension of the cables for the coordination. Recently, a system comprising a bar tethered to two heterogeneous drones was presented [5]. While cooperative aerial transportation with towed-cables [4], [5] has been a popular and practical method, human operators are required to manually tie the desired object to the cables. This requirement may prevent the applications of the method in risky terrains. However, aerial manipulation with multi-degree-of-freedom (multi-DOF) robotic arm [13] can help lift and transport a payload on its own.

To employ cooperative aerial manipulation with a multi-DOF robotic arm for effective transportation, several problems associated with multiple aerial robots, such as those related to safety and complexity should be resolved. For addressing the complexities of rigidly attached aerial robots, researchers have developed a coordination algorithm [6]–[8]. In [6], the coordination problem was solved using optimization-based force decomposition. In [7], a formation control method for cooperatie robots based on constrained optimization was presented for obstacle environments. The authors proposed a local motion planner for an optimal formation and a global planner based on a sampling-based planning algorithm. The formation-based method was demonstrated in two cases: a team of aerial robots flying in formation and a team of mobile manipulators collaboratively carrying an object. Despite satisfactory results, their method involves a complex nonlinear optimization approach, which requires heavy computational load. In [8], the authors proposed a coordinated motion algorithm based solely on inverse kinematics for aerial robotic manipulators and showed satisfactory experimental results without using an actual payload. However, this algorithm [8] was verified only for an obstacle-free environment, and therefore, it is difficult to apply it to velocity-constrained aerial manipulation in obstacle environments.

To solve the path planning problem for cooperative aerial manipulators in obstacle environments, the authors of [9], [10], [14] applied a sampling-based planning algorithm such as a rapidly-exploring random tree (RRT). Since RRT does not depend on the explicit representation of obstacles, it is widely applied to various motion planning problems in obstacle environments [15]. In [9], [14], an online motion generation algorithm was developed for an obstacle environment by exploiting RRT*, which is an extended version of RRT, and dynamic movement primitives (DMPs). In [10], the motion planning problem was formulated for an aerial platform with the dual-arm manipulator and solved with RRT* algorithms to optimize the performance of the trajectories in terms of energy and time. However, obtaining the optimal path with RRT* in offline motion requires a higher computational load. To overcome this requirement, our previous study [11] proposed motion generation with a combination of inverse kinematics and DMPs for obstacle environments. However, these studies [9]–[11], [14] did not consider the velocity and curvature constraints.

A potential-based obstacle avoidance algorithm for cooperative manipulators was presented in [16], [17]. The potential field method is a popular obstacle avoidance method for mobile robots [18], drones [19], and multiple underwater vehicles [16]. In [17], the desired formation for cooperative aerial manipulators was realized using a distributed controller, while collision avoidance was guaranteed by using an artificial potential function (APF) method. Since the use of the conventional APF gives rise to the local minima problem and sudden changes in the desired path, a harmonic potential function (HPF)-based approach has been presented for cooperative mobile robots [20]. Although smooth trajectories were generated in obstacle environment without performing any complex optimization or using a sampling method, this approach was verified only for the formation control of multiple mobile robots. Moreover, the velocity and curvature constraints cannot be considered in the approach. For the consideration of velocity and curvature constraints, an approach based on streamline-based HPF was proposed in [21], but the approach was designed for a single mobile robot. In addition, since those algorithms [20], [21] were designed for a mobile robot without a rigid grasp, additional studies are required to adapt these algorithms for robots with a rigid grasp.

## II. DECENTRALIZED DYNAMICS

In this section, we discuss the dynamics of cooperative aerial manipulators with a multi-DOF robotic arm. Detailed kinematic relations and equations of motion can be found in our previous work [11]. The coordinated frames $\Sigma_I$, $\Sigma_{b,i}$, and $\Sigma_{c,i}$ denote the inertial frame, body frame of the multirotor, and body frame of the end-effector for the $i$-th aerial manipulator, respectively (see Fig. 1). $\Sigma_o$ represents the body frame of a common payload.

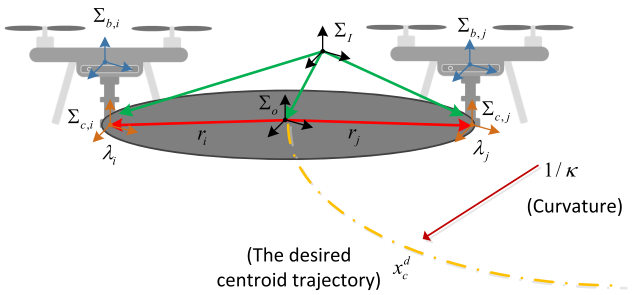



**FIGURE 1.** Two cooperative aerial robots, each consisting of a multirotor and a robotic arm, manipulate a common object.

### A. AERIAL MANIPULATOR AND PAYLOAD DYNAMICS

Using the position of the center of mass of the multirotor in $\Sigma_I$, we define the states of the $i$-th manipulator $\mathbf{q}_i \in \mathbb{R}^{6+n}$ $(i = 1, \ldots, N)$ in terms of the position of the multirotor, Euler angle of the multirotor and joint angles of a robotic arm. Here, $n$ is the number of degree of freedom (DOFs) of a robotic arm and $N$ represents the total number of aerial manipulators.

When an aerial manipulator interacts with a payload, the wrench $\boldsymbol{\lambda}_i \in \mathbb{R}^6$ is applied to the end-effector of the $i$-th aerial manipulator in $\Sigma_{c,i}$ (See Fig. 1). In this case, the dynamics of the $i$-th the aerial manipulator can be obtained as

$$M_i(\mathbf{q}_i)\ddot{\mathbf{q}}_i + Q_i(\mathbf{q}_i, \dot{\mathbf{q}}_i)\dot{\mathbf{q}}_i + W_i(\mathbf{q}_i) = \boldsymbol{\tau}_i - J_i^T(\mathbf{q}_i)\boldsymbol{\lambda}_i, \quad (1)$$

where $J_i(\mathbf{q}_i) \in \mathbb{R}^{6\times 8}$ means the Jacobian matrix from $\Sigma_{b,i}$ to $\Sigma_{c,i}$. $M_i \in \mathbb{R}^{8\times 8}$ is the inertia matrix, $Q_i \in \mathbb{R}^{8\times 8}$ is the Coriolis matrix, and $W_i \in \mathbb{R}^8$ is the gravity term. $\boldsymbol{\tau}_i$ is the control input in the inertial frame $\Sigma_I$.

The dynamics of a payload can be defined as

$$H_o\ddot{\mathbf{q}}_o + \mu_o\dot{\mathbf{q}}_o + G_o = \sum_{i=1}^{N} E_i\boldsymbol{\lambda}_i, \quad (2)$$

where $\dot{\mathbf{q}}_o \in \mathbb{R}^6$ is the twist of a payload. The twist is represented as a six-dimensional vector comprising the translational velocity and rotational velocity of the payload in $\Sigma_I$. We use $H_o = \mathrm{diag}(m_oI_3, J_o) \in \mathbb{R}^{6\times 6}$, where $m_o$ is the mass of a payload and $J_o \in \mathbb{R}^{3\times 3}$ is the inertia, $I_3$ is $3 \times 3$ identity matrix. The parameter $\mu_o \in \mathbb{R}^{6\times 6}$ is a matrix containing the Coriolis and centripetal terms and $G_o \in \mathbb{R}^6$ is a gravity matrix. $E_i \in \mathbb{R}^{6\times 6}$ is a grasp matrix and is given by

$$E_i = \begin{bmatrix} I_3 & 0_3 \\ S(\mathbf{r}_i) & I_3 \end{bmatrix}, \quad (3)$$

where $0_3$ is a $3 \times 3$ zero matrix and $S(\mathbf{r}_i)$ is the skew-symmetric matrix expressing the cross product from the position of $\Sigma_o$ with respect to $\Sigma_{c,i}$.

### B. COMBINED DYNAMICS

With the assumption of a rigid grasp, the effective wrench $\boldsymbol{\lambda}_i$ applied by the $i$-th aerial manipulator can be calcluated using the force distribution solution [22] as

$$\boldsymbol{\lambda}_i = c_iE_i^{\dagger}(H_o\ddot{\mathbf{q}}_o + \mu_o\dot{\mathbf{q}}_o + G_o), \quad (4)$$

where and $E_i^{\dagger}$ can be obtained using the Moore-Penrose pseudo-inverse solution as

$$c_iE_i^{\dagger} = \begin{bmatrix} c_iI_3 & -c_iS(\mathbf{r}_i)\Pi^{-1} \\ 0_3 & c_i\Pi^{-1} \end{bmatrix},$$

with $\Pi = I_3 + \sum_{i=1}^{N} c_iS(\mathbf{r}_i)S^T(\mathbf{r}_i)$. For simplicity, we assume that all aerial manipulators are homogeneous. Therefore, they contribute torque of equal magnitude at the center of the payload in $\Sigma_o$. In this case, we set $c_i = 1/N$.

Since all positions and orientations of the common payload and the end-effectors coordinates can be expressed relative to a common reference frame, we can obtain the following equation between $\mathbf{q}_i$ and $\mathbf{q}_o$ as

$$\dot{\mathbf{q}}_o = E_i^{-T}J_i\dot{\mathbf{q}}_i. \quad (5)$$

By substituting (4) and (5) into (1), we can express the decentralized equation of motion of the $i$-th aerial manipulator with the payload as

$$D_i(\mathbf{q}_i)\ddot{\mathbf{q}}_i + C_i(\mathbf{q}_i, \dot{\mathbf{q}}_i)\dot{\mathbf{q}}_i + G_i(\mathbf{q}_i) = \boldsymbol{\tau}_i. \quad (6)$$

Here, the matrices are calculated as

$$\begin{aligned} D_i &= M_i(\mathbf{q}_i) + c_iM_o(\mathbf{q}_i),\ G_i = W_i + c_iW_o(\mathbf{q}_i) \\ C_i &= Q_i(\mathbf{q}_i, \dot{\mathbf{q}}_i) + c_iQ_o(\mathbf{q}_i, \dot{\mathbf{q}}_i) + c_iJ_i^T(E_i^{\dagger}H_oE_i^{-T})\dot{J}_i \end{aligned} \quad (7)$$

where the following representations hold

$$\begin{aligned} M_o(\mathbf{q}_i) &= J_i^T(E_i^{\dagger}H_oE_i^{-T})J_i \in \mathbb{R}^{8\times 8}, \\ Q_o(\mathbf{q}_i, \dot{\mathbf{q}}_i) &= J_i^T(E_i^{\dagger}\mu_oE_i^{-T})J_i \in \mathbb{R}^{8\times 8}, \\ W_o(\mathbf{q}_i) &= J_i^TE_i^{\dagger}G_o \in \mathbb{R}^8. \end{aligned} \quad (8)$$

### C. ACTUATION COMMANDS

In the case of cooperative aerial manipulators as shown in Fig. 1, the control input $\boldsymbol{\tau}_i$ can be converted into an the actuation command as:

$$\boldsymbol{\tau}_i = \Xi_i\mathbf{f}_i \quad (9)$$

where $\mathbf{f}_i = [\mathbf{h}_i^T, \boldsymbol{\tau}_{\eta,i}^T]^T$ in $\Sigma_{b,i}$ consists of $\mathbf{h}_i$ for the input force command of the $i$-th multirotor and $\boldsymbol{\tau}_{\eta,i}$ for the command to the arm. The parameter $\Xi_i \in \mathbb{R}^{(6+n)\times(6+n)}$ converts $\boldsymbol{\tau}_i$ into $\mathbf{f}_i$ and includes the motor mapping matrix which depends on the arm length of the multirotor, drag coefficient $(k_m)$ and thrust coefficient $(k_f)$ of motors . Since $\Xi_i^T\Xi_i$ is always invertible except when the pitch angle is a multiple of $\pm k\frac{\pi}{2}$, we can obtain the desired thrust for each motor and the robotic arm [23] as:

$$\mathbf{f}_i = \Xi_i^{\dagger}\boldsymbol{\tau}_i \quad (10)$$

where $^{\dagger}$ denotes the Moore-Penrose pseudo-inverse. In this case, since the thrust of the $k$-th motor can be computed as $h_i(k) = k_f(\omega_k^d)^2$ by using the rotational velocity of the $k$-th rotor $\omega_k^d$, we can provide the speed command for each motor as $\omega_k^d = \sqrt{h_i(k)}/k_f$. However, since a motor has a limited maximum rotational speed as

$$\omega_k^d < \omega_{max}^d, \quad (11)$$

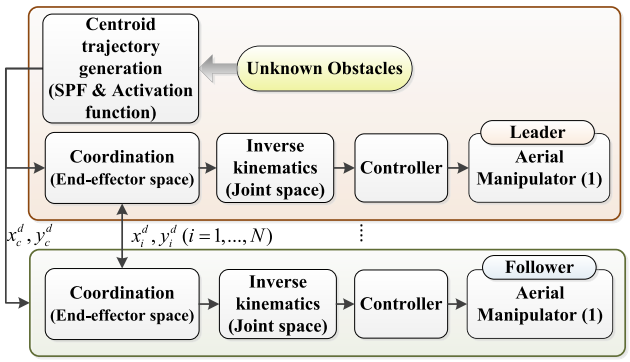



**FIGURE 2.** Structure of the proposed method.

where $\omega_{max}^d$ denotes the maximum rotational speed, it is important to plan the desired trajectory so that it satisfies this condition. Because of the vibrations of a multirotor, in general, the actual maximum rotor speed $\omega_k^d$ is much less than $\omega_{max}^d$, and therefore the planned trajectory should require minimal changes to the motor. In addition, if the desired trajectory of cooperative drones has sharp turns, then the interaction forces can be increased. This is mainly because a sharp turn requires a higher acceleration to follow, whose value depends on the curvature of the desired trajectory.

In this study, we considered path planning by applying constraints on the motor speed and curvature constraint to reduce the interaction forces. After discussing the proposed algorithm in Section III, we present simulation (Section IV) and experimental results (Section V) obtained with the algorithm.

## III. A PLANNING FRAMEWORK

In this section, we propose a path planning framework for cooperative aerial manipulators based on the decentralized dynamics in (6). The structure of the proposed method is shown in Fig. 2. The leader manipulator which detects obstacles and generates an avoidance trajectory for the centroid of the cooperative aerial manipulators under velocity and curvature constraints. Subsequently, the follower and leader robots calculate the trajectory of their own end-effector under velocity constraints. The desired trajectory of the end-effectors (i.e., $\mathbf{q}_{e,i}^d \in \mathbb{R}^3$) can be converted into a trajectory in joint space (i.e., $\mathbf{q}_i^d \in \mathbb{R}^{6+n}$) by exploiting inverse kinematics.

### A. HARMONIC POTENTIAL FUNCTION

To solve the problems of the APF including the local minima problem and generation of a sharp trajectory near obstacles, HPF-based planning has been proposed. In this subsection, we briefly explain the basic formulation of a streamline-based HPF on the basis of the description in [24]. In a two-dimensional plane, the harmonic function $\phi$ can be computed by solving Laplace's equation $\nabla^2\phi = 0$ with a boundary condition that governs the flow of an incompressible inviscid fluid motion at every point. The velocity potential can be defined in the complex plane $\mathbb{C}^2$ as

$$f(z) = \phi(x, y) + j\psi(x, y), \tag{12}$$

where $z = x + jy$, $j$ is the imaginary unit, and $\psi(x, y)$ and $\phi(x, y)$ are the stream function and the velocity potential, respectively. The complex velocity is computed by differentiating the velocity potential with respect to $z$ as

$$\frac{df(z)}{dz} = \dot{x}(z) - j\dot{y}(z), \tag{13}$$

where $\dot{x}(z)$ and $\dot{y}(z)$ are the velocity along the $x$ and $y$ axis, respectively.

In this study, we consider an attractive function for goal navigation and the obstacle potential function for obstacle avoidance. The obstacle potential function $f_{cyn}(z)$ for a circular obstacle and $f_{sk}$ for sinking to the goal can be defined as

$$f_{cyn}(z) = \frac{r^2}{z}, \quad f_{sk}(z) = c_s ln(z), \tag{14}$$

where $c_s$ is the user-defined control gain and $r$ is determined by the sizes of the obstacle and robot. Note that the principle of superposition is used to generate the flow of multiple elements.

The difference between the streamline-based HPF in (14) and the conventional APF-based obstacle avoidance technique proposed in [25] is shown in Fig. 3. Clearly, the streamline-based HPF generates a more smooth path.

### B. PLANNING UNDER VELOCITY AND CURVATURE CONSTRAINTS

Before deriving the planning algorithm, we regulate the state variables for the proposed method. Let us consider a situation where two drones try to avoid each other. For cooperative aerial manipulators in the decentralized control mode based on eq. (6), it is difficult to modify the preset height difference between the manipulators because of the constraints imposed by the rigid grasp in (4). In addition, the goal is carefully selected by making the following assumption.

*Assumption 1:* The desired altitude has already been chosen to avoid an obstacle with a large horizontal extent.

Accordingly, for simplicity, we consider only the desired trajectory in the horizontal plane, $\mathbf{x}_c^d = [x_c^d, y_c^d]^T$ for the $i$-th aerial manipulator to avoids the vertical obstacles. Here, the superscript $d$ denotes the desired value. Note that the Assumption 1 can be relaxed by using a proper height planning algorithm such as the RRT, or a using different 2-dimensional plane such as $[x_c, z_c]^T$. Fig. 4 shows the detailed structure of the centroid trajectory generation and the coordination in Fig. 2.

Given the Cartesian position of an obstacle $\mathbf{x}_k = [x_k, y_k]^T$ and a target point $\mathbf{x}_f = [x_f, y_f]^T$, we define new variables

$$\begin{aligned} z_{ck} &= (x_c^d - x_k) + j(y_c^d - y_k) := x_{ck} + jy_{ck} \\ z_{cf} &= (x_c^d - x_f) + j(y_c^d - y_f) := x_{cf} + jy_{cf}. \end{aligned} \tag{15}$$

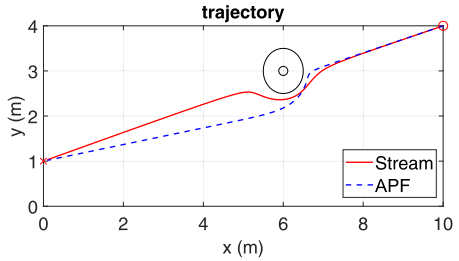



**FIGURE 3.** Comparison between APF and SPF.

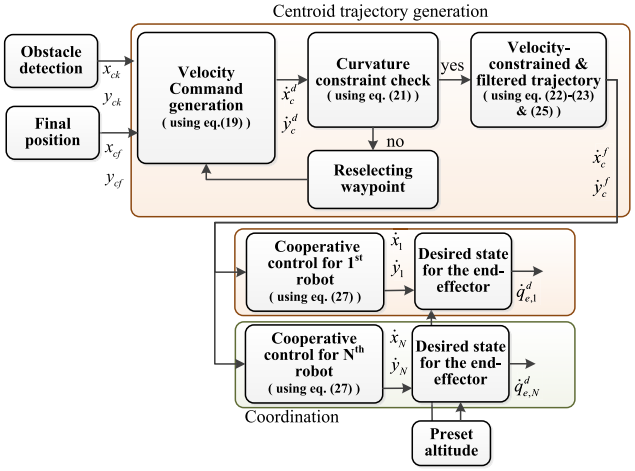



**FIGURE 4.** Detailed structure of the centroid trajectory generation and the coordination.

From the obstacle potential function $f_{cyn}(z_{ck})$ in eq. (14), we can obtain the following equation

$$\frac{df_{cyn}(z_{ck})}{dz_{ck}} = -\frac{r^2}{z^2} = -\frac{r^2}{(x_{ck}+jy_{ck})^2} = -\frac{r^2(x_{ck}^2 - j2x_{ck}y_{ck} - y_{ck})^2}{(x_{ck}^2+y_{ck})^2}. \quad (16)$$

Using the target point eq. (15), the sinking potential can be calculated as

$$\frac{df_{sk}(z_{cf})}{dz_{cf}} = \frac{-c_s}{x_{cf}+jy_{cf}} = \frac{-c_s(x_{cj}-jy_{cf})}{x_{cf}^2+y_{cf}^2}. \quad (17)$$

In this method, we use the normalized sinking potential function because of the initial and final velocity condition which we will discuss later. Therefore, we can compute the navigation velocity command for the target point ($U_x$ in $x$ direction and $U_y$ in $y$ direction) based on eq. (17) as

$$U_x = \frac{-c_s x_{cf}}{(x_{cf}^2+y_{cf}^2)^{3/2}},\ U_y = \frac{-c_s y_{cf}}{(x_{cf}^2+y_{cf}^2)^{3/2}}. \quad (18)$$

Then the velocity command can be computed by the superposition rule with eq. (16) and eq. (18) as

$$\dot{x}_c^d = U_x - \frac{2A(x_{ck}^2-y_{ck}^2)}{(x_{ck}^2+y_{ck}^2)^2},\ \dot{y}_c^d = U_y - \frac{2Ax_{ck}y_{ck}}{(x_{ck}^2+y_{ck}^2)^2} \quad (19)$$

where $A = Ur^2$ and $U = \sqrt{U_x^2+U_y^2}$ is the velocity in longitudinal direction. To calculate the curvature of the desired trajectory, the acceleration $a_x$ and $a_y$ are required with respect to $\mathbf{x}_c^d$, which can be written as

$$a_x^d = \frac{\delta \dot{x}_c^d}{\delta x_c^d}\dot{x}_c^d + \frac{\delta \dot{x}_c^d}{\delta y_c^d}\dot{y}_c^d,\ a_y^d = \frac{\delta \dot{y}_c^d}{\delta x_c^d}\dot{x}_c^d + \frac{\delta \dot{y}_c^d}{\delta y_c^d}\dot{y}_c^d, \quad (20)$$

where

$$\frac{\partial \dot{x}_c^d}{\partial x_c^d} = \frac{c_s(x_{cf}^2-y_{cf}^2)}{(x_{cf}^2+y_{cf}^2)^{5/2}} + \frac{4Ax_{ck}(x_{ck}^2-3y_{ck}^2)}{(x_{ck}^2+y_{ck}^2)^3}$$

$$\frac{\partial \dot{x}_c^d}{\partial y_c^d} = \frac{2c_s x_{cf}y_{cf}}{(x_{cf}^2+y_{cf}^2)^{5/2}} - \frac{4Ay_{ck}(x_{ck}^2-3y_{ck}^2)}{(x_{ck}^2+y_{ck}^2)^3}$$

$$\frac{\partial \dot{y}_c^d}{\partial x_c^d} = \frac{2c_s x_{cf}y_{cf}}{(x_{cf}^2+y_{cf}^2)^{5/2}} + \frac{2Ay_{ck}(4x_{ck}y_{ck}-x_{ck}^2-y_{ck}^2)}{(x_{ck}^2+y_{ck}^2)^3}$$

$$\frac{\partial \dot{y}_c^d}{\partial y_c^d} = \frac{-c_s(x_{cf}^2-y_{cf}^2)}{(x_{cf}^2+y_{cf}^2)^{5/2}} + \frac{2Ax_{ck}(4x_{ck}y_{ck}-x_{ck}^2-y_{ck}^2)}{(x_{ck}^2+y_{ck}^2)^3}.$$

Note that $\partial x_{ck}/\partial x_c^d = \partial x_{cf}/\partial x_c^d = 1$. Finally, using (20), we can compute the curvature of the trajectory as

$$\kappa = \frac{\dot{x}_c^d a_y^d - \dot{y}_c^d a_x{}^d}{[(\dot{x}_c^d)^2+(\dot{y}_c^d)^2]^{3/2}}. \quad (21)$$

The curvature is calculated as $1/\kappa$, and therefore, a small $\kappa$ yields paths similar to a straight line.

Now, let us consider the velocity constraint. The equation (19) gives commands for navigation to the target $\mathbf{x}_f$, but this command cannot satisfy the velocity and curvature constraints. To resolve this problem, first, we design a velocity command for the velocity constraint:

$$\dot{\mathbf{x}}_c^v = \begin{cases} v_{\max}\dfrac{\dot{\mathbf{x}}_c^d}{\|\dot{\mathbf{x}}_c^d\|} & \text{if } u \geq u_{\max} \\ \dot{\mathbf{x}}_c^d & \text{otherwise,} \end{cases} \quad (22)$$

where $v_{max} \in \mathbb{R}$ is the maximum velocity. Since the conventional potential field approach [25] can lead to the drones to lose their direction when the path is normalized because of velocity commands that cause large velocity changes, it is difficult to apply the velocity constraints in (22). However, the streamline-based HPF can help the drones maintain their direction after normalization (22). This technique can guarantee that the desired trajectory will approach the target point when the velocity is below $v_{max}$. Nevertheless, in this technique, the initial and final velocities are sufficiently large because of normalization, resulting in an increase in the acceleration. Accordingly, we define the activation function for the navigation function in eq. (14) as

$$c_s = \begin{cases} c_s & \text{if } \|x - x_f\| > b_p \\ c_s \times h_g(\dfrac{\|x-x_f\|}{b_p}) & \text{otherwise,} \end{cases} \quad (23)$$

where $h_g(a) = 6x^5 - 15x^4 + 10x^3$, which satisfies $h(0) = 0$, $h(1) = 1$, $\dot{h}(0) = 0$ and $\dot{h}(1) = 0$. The parameter $b_p$

**Algorithm 1** VCC Trajectory Generation Algorithm

1: **Given** :$\mathbf{x}_c^{ini}$, $\mathbf{x}_f$
2: **while** $\left\|\mathbf{x}_c^d - \mathbf{x}_f\right\| > 0$ **do**
3: $\quad\mathbf{u}_{xy} \leftarrow$ using eq. (19), (23) and (24);
4: $\quad\mathbf{x}_c^d \leftarrow$ `Trajectory`$(\mathbf{R}_c^d, \mathbf{x}_c^{ini})$
5: $\quad$**if** `NewObstacle`$(\mathbf{x}_k)$ **then**
6: $\quad\quad\mathbf{x}_{in} \leftarrow \mathbf{x}_c^d$; `flag` $\leftarrow 1$;
7: $\quad\quad$**while** `flag` **do**
8: $\quad\quad\quad\mathcal{T}_{in} \leftarrow$ `Trajectory`$(\mathbf{R}_c^d u, \mathbf{x}_{in}, \mathbf{x}_k)$
9: $\quad\quad\quad\kappa \leftarrow FindMax_\kappa(\mathcal{T}_{in})$
10: $\quad\quad\quad$**if** $\kappa > \kappa_{max}$ **then**
11: $\quad\quad\quad\quad$**if** `PlaneCheck`$(\mathbf{x}_{in}, \mathbf{x}_k, \mathbf{x}_f)$ **then**
12: $\quad\quad\quad\quad\quad\mathbf{x}_{in} = \mathbf{x}_{in} + R_o^I[0, step]^T$;
13: $\quad\quad\quad\quad$**else**
14: $\quad\quad\quad\quad\quad\mathbf{x}_{in} = \mathbf{x}_{in} - R_o^I[0, step]^T$;
15: $\quad\quad\quad\quad$**end if**
16: $\quad\quad\quad$**else**
17: $\quad\quad\quad\quad\mathbf{x}_c^d \leftarrow \mathbf{x}_{in}$, `flag` $\leftarrow 0$;
18: $\quad\quad\quad$**end if**
19: $\quad\quad$**end while**
20: $\quad$**end if**
21: $\quad\mathbf{x}_c^f \leftarrow$ `Filter`$(\mathbf{x}_c^d)$
22: **end while**
23: Return$\mathbf{x}_c^f$

is a user-defined constant, which represents a region of the increasing or decreasing velocity. Near the target point $\mathbf{x}_f$, $c_s$ converges 0 gradually. If $c_s$ is zero, the velocity command will give zero value. Unlike the target point, $c_s$ increase from a positive constant $\epsilon > 0$ to $c_s$ near the starting point. This is because the aerial robot will never leave the starting point if the velocity command remains zero because of $c_s = 0$. Therefore, the function $h_g$ will be replaced by $h_s$ near the starting point:

$$h_s(a) = \frac{1}{1+\epsilon}(6a^5 - 15a^4 + 10a^3 + \epsilon). \tag{24}$$

Here the function $h_s(a)$ satisfies $h_s(0) = \frac{\epsilon}{1+\epsilon}$, $h_s(1) = 1$, $\dot{h}_s(0) = 0$, and $\dot{h}_s(1) = 0$. Therefore, $c_s$ varies from $\frac{\epsilon c_s}{1+\epsilon}$ to $c_s$ near the starting point, which prevents the robot from remaining at the starting point.

We next discuss curvature constraints. The curvature should be maximized to reduce the internal forces of cooperative aerial manipulators, and the maximization corresponds to finding the condition $\kappa < \kappa_{\max}$. In Algorithm 1, we provide a pseudocode for velocity and curvature-constrained (VCC) trajectory generation based on eq. (21). Given the initial point (i.e., $\mathbf{x}_c^{ini}$) and the final point (i.e., $\mathbf{x}_f$), the velocity command is computed using eqs. (19), (23) and (24) (Line 3). Using the command, the algorithm generates the desired trajectory (i.e., $\mathbf{x}_c^d$ , Line 4). If a new obstacle is detected (Line 5) while following the desired trajectory, the algorithm saves the current location (i.e., $\mathbf{x}_{in}$) and changes the boolean variable `flag`(Line 6). Subsequently, a possible trajectory ($\mathcal{T}_{in}$) with respect to $\mathbf{x}_{in}$ is calculated (Line 8). After checking the possibility of collision with an obstacle for $\mathcal{T}_{in}$, the algorithm finds the maximum value of $\kappa$ for $\mathcal{T}_{in}$ (Line 9). If $\kappa > \kappa_{max}$, the saved current location $\mathbf{x}_{in}$ is changed to satisfy the curvature constraints (Line 11-18). On the basis of $\mathbf{x}_f$, $\mathbf{x}_k$, and $\mathbf{x}_{in}$, the `PlaneCheck` function determines the direction of motion for avoiding obstacles (e.g., going right or left). On the basis of $R_o^I$, which denotes the rotation matrix of the relative direction between a robot and an obstacle, the waypoint $\mathbf{x}_{in}$ is modified as shown in Fig. 5 (Lines 12 and 14).

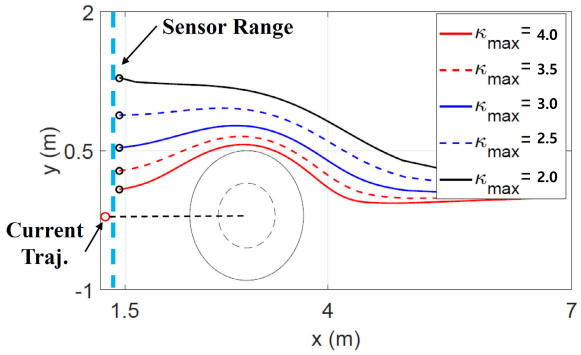



**FIGURE 5.** Concept of waypoint selection algorith.

To avoid the sudden changes in the desired trajectory, a filtered trajectory was finally selected as the desired trajectory (i.e., $\mathbf{x}_c^f$). In the `Filter` function (Line 21), we used a moving average filter and a low-pass filter. In the desired trajectory, since our VCC trajectory generation algorithm reselects the waypoint to satisfy the curvature constraints $\kappa_{max}$, sudden changes in the waypoints could occur, as shown in Fig. 5. If the newly-selected waypoint is far away from the entry waypoint, the curvature of the desired trajectory may increase. To prevent any such increase, we perform additional path smoothing with a moving-average filter and a low-pass filter. In our algorithm, we use trajectory data during a certain discrete time step for applying the moving average filter, and subsequently, the following low-pass filter was applied

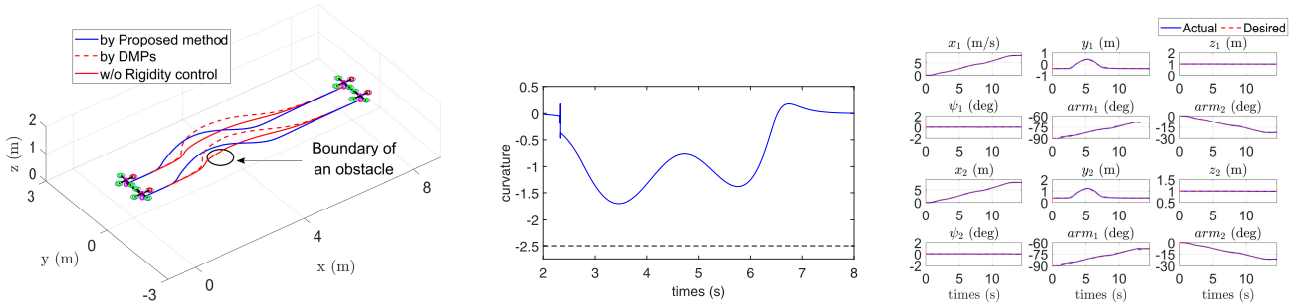
$$\alpha\dot{\mathbf{x}}_c^f + \mathbf{x}_c^f = \mathbf{x}_c^m, \mathbf{x}_c^f(0) = \mathbf{x}_c^m(0) \tag{25}$$

where $\mathbf{x}_c^m$ is the filtered output of the moving-average filter and $\alpha$ is a user-defined time constant. Note that if the time constant is too large, the desired trajectory may be severely distorted, resulting in collision with obstacles along the path. Due to such reason, we select $\alpha$ less than 0.5. The results of this filter are discussed in the simulation section.

### C. COOPERATIVE CONTROL

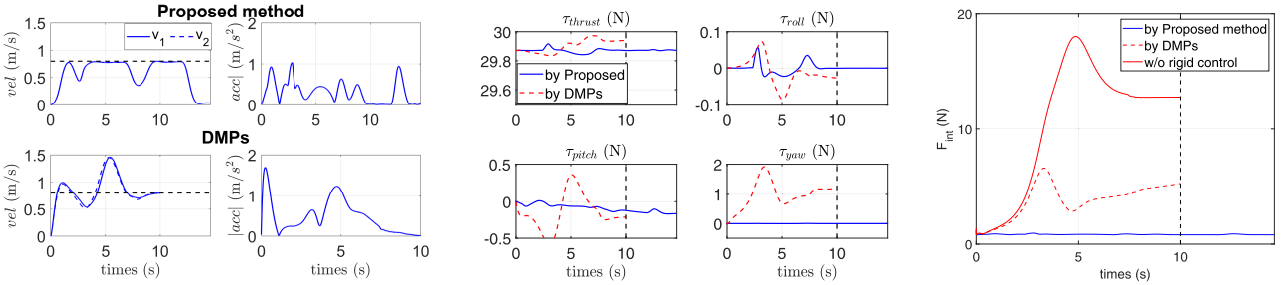
Our algorithm considers the sizes of the cooperative robots and the size of obstacles to define $r$ in eq. (16), and it generates the filtered trajectory of a centroid (i.e., $\mathbf{x}_c^f$. By using $\mathbf{x}_c^f$, each aerial manipulator computes its trajectory to maximize the following cost function

$$J = \sum_{k=1(i\neq k)}^{N} (\left\|\mathbf{x}_i^d - \mathbf{x}_k^d\right\|^2 - d_{ik})^2 + \left\|\frac{\mathbf{x}_i^d + \mathbf{x}_k^d}{2} - \mathbf{x}_c^f\right\|. \tag{26}$$




(a) Collision avoidance scenario (b) Curvature of the path (C) Time histories of $\mathbf{q}_i^d$

**FIGURE 6.** **Simulation scenario.**




(a) Velocity constraint (b) Control inputs (c) Internal force during the flight

**FIGURE 7.** **Comparison results.**

Here $i \neq k$. To find the $\mathbf{x}_i^d = [x_i^d, y_i^d] \in \mathbb{R}^2$ for each aerial manipulator, we use the negative gradient as

$$\dot{\mathbf{x}}_i^d = -\frac{\delta J}{\delta \mathbf{x}_i^d}, \tag{27}$$

where $d_{ij}$ is the desired distance between the grasping points of the manipulators. As evident in eqs. (26) and (27), the aerial manipulator maintains the desired distance $d_{ij}$ while the current centroid trajectory (i.e., $(\mathbf{x}_i + \mathbf{x}_j)/2$) follows the desired centroid trajectory $\mathbf{x}_c^f$.

Since $\mathbf{x}_i^d$ is the trajectory in the end-effector frame as shown in Fig 2, the trajectory should be converted to the desired trajectory in joint space ($\mathbf{q}_i^d$). For this, we first define the following desired trajectory of the end-effector:

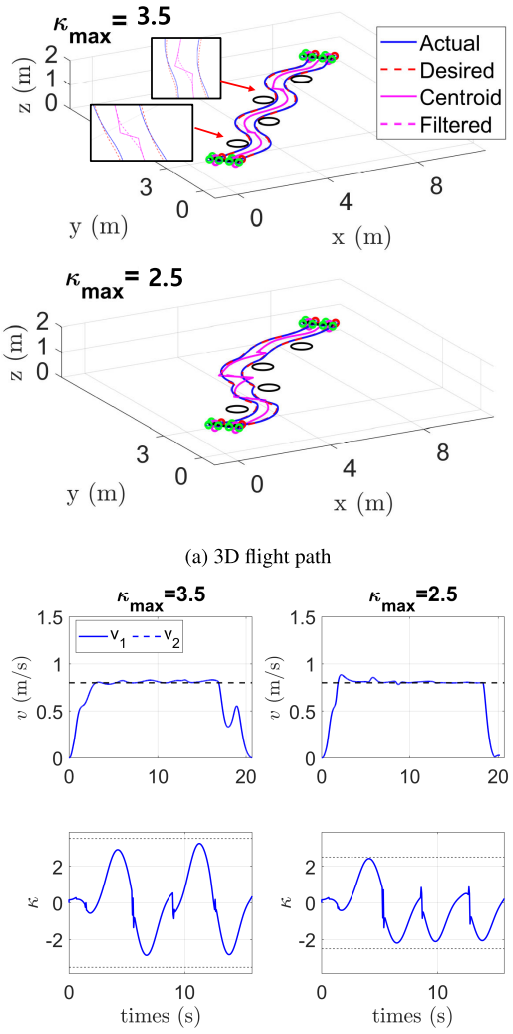
$$\mathbf{q}_{e,i}^d = [x_i^d, y_i^d, z_i^d, cos(\theta_o)] \in \mathbb{R}^4, \tag{28}$$

where $x_i^d$ and $y_i^d$ are obtained from our proposed planning algorithm. In $\mathbf{q}_{e,i}^d$, there exist additional terms including $z_i^d$ and $cos(\theta_o)$. Here, $z_i^d$ is the desired altitude, $\theta_o$ is the desired attitude of the payload and $cos(\theta_o)$ is the term representing the desired attitude of the end-effector, which is essential for applying the rigid grasping method. By adjusting $\theta_o = \frac{\pi}{2}$, we can change the direction of the end-effector. For example, if $\theta_o = \frac{\pi}{2}$, the end-effector will face the downward direction. This adjustment is required when an aerial manipulator grasps or releases a payload. Therefore, we can set $cos(\theta_o) = 1$ in our scenario which concentrates on transporting a payload after the manipulator grasps an object. Finally, $\mathbf{q}_{e,i}^d$ is converted into $\mathbf{q}_i^d$ by exploiting inverse kinematics. Details of the inverse kinematics are provided in our previous work [11].

## IV. SIMULATION

Before discussing the experimental validation of the proposed method, two simulation results are presented to show the performance of the proposed method. In both simulations, each aerial manipulator consists of a quadrotor and a 2-DOF arm. In the first simulation, we compare our proposed method with DMPs in [11] to analyze the internal stability during cooperative flight. In the second simulation, two different types of curvature constraints are considered in multiple obstacle environments.

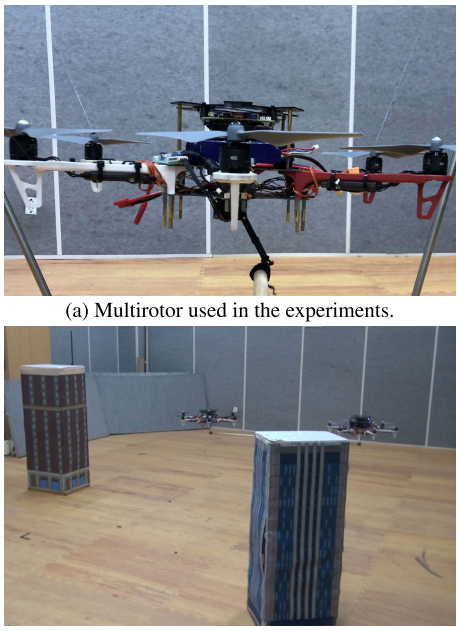
We compare our proposed method with DMPs for a single obstacle environment with $v_max = 0.8$ m/s and $\kappa_{max} < 2.5$. Fig. 6(a) shows the 3D flight path obtained with the proposed method. The curvature of the generated path and the state tracking performance in joint space (i.e., $\mathbf{q}_i$ and $\mathbf{q}_i^d$) are shown in Fig. 6(b)-(c). Fig. 7 shows the comparison results for the velocity constraint, control input, and internal forces, respectively. While the final arrival time for our method is longer than that for DMPs, our method can satisfy the velocity constraint without requiring any computationally-expensive optimization algorithm. Besides, since the desired path obtained with our method is smoother than that derived using DMPs, the control inputs are smaller than those for




(a) 3D flight path

(b) velocity and curvature constraints

**FIGURE 8.** Simulation in obstacle environments.



(a) Multirotor used in the experiments.

(b) Picture taken during flight.

**FIGURE 9.** Experimental setup.

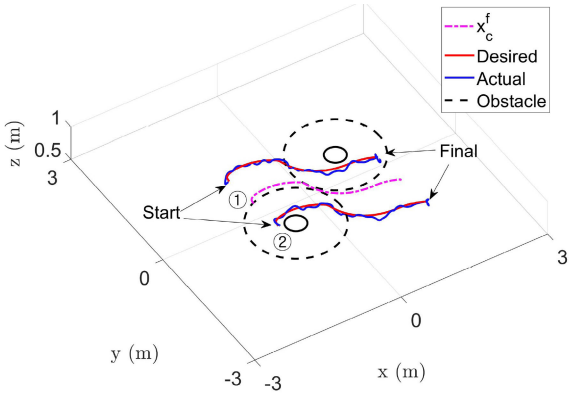



**FIGURE 10.** Three-dimensional trajectory tracking performance.

DMPs. The control inputs are closely related to the maximum motor speed. The comparison of the internal force is shown in Fig. 7(c). The red line denotes the trajectory when an aerial manipulator avoids obstacles, but another aerial manipulator wants to maintain the original desired trajectory. This is the case when the internal force during flight is maximum. The red dotted line is the trajectory obtained with DMPs. Unlike the aforementioned method, DMPs modify their desired trajectory to reduce the internal force. However, a large internal force is generated near obstacles, and this force should be reduced for internal stability. By employing our method along with an on-line path smoothing method, we can reduce the internal force near obstacles to achieve a more stable system.

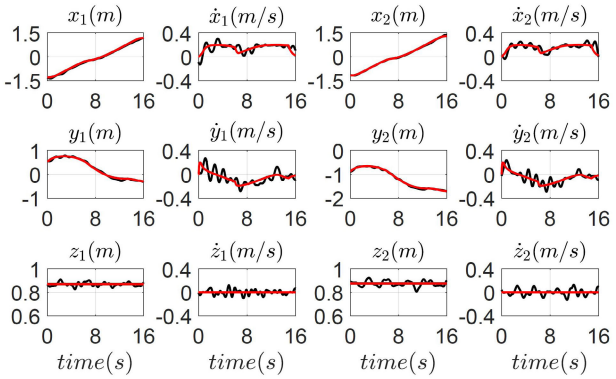
Fig. 8 shows the planning results in the multiple obstacle environments under velocity and curvature constraints. The velocity constraint is 0.8 m/s, and $\kappa_{max} < 3.5$ or $\kappa_{max} < 2.5$. In Fig. 8(a), the magenta line denotes the centroid trajectory generated by Algorithm 1, the red dotted line is the desired trajectory for each aerial manipulator, and the blue line is the actual trajectory. Fig. 8(b) shows the actual velocity and curvature during flights. For filter, in the simulation, we used trajectory data during 0.5 seconds, and we set $\alpha = 0.5$ for the low-pass filter. As evident in Fig. 8(a), the filtered trajectory is smoothed by our proposed method.

## V. EXPERIMENTS

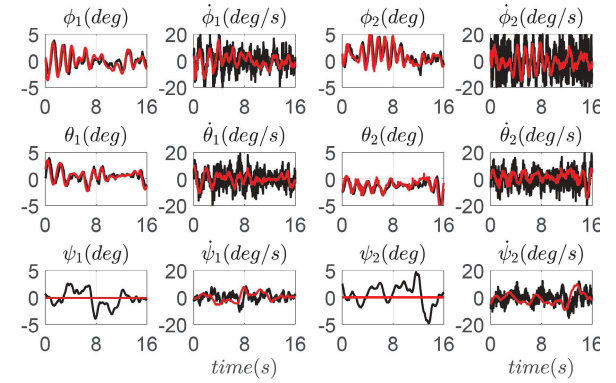
In this section, the experimental results are described. In the experiment, two custom-made aerial manipulators were used to carry an unknown payload.

### A. HARDWARE SETUP

The multirotor used in this study is based on the DJI F550-based platform, and it is equipped with 6 DJI E300 motors. For the indoor flight test, we use a motion capture system (Vicon). The Vicon measures the position information at 100 Hz and transmits it to an on-board computer of the aerial manipulator through a Crazyradio PA module. The on-board




(a) Position tracking performance




(b) Attitude tracking performance

**FIGURE 11. Position and attitude tracking performance.**

compuer was Intel NUC7i7BNH with a 3.5GHz Intel Core i7-7567U processor. Furthermore, a Pixhawk autopilot was attached to the center of the multirotor and connected to the onboard computer via universal serial bus (USB). The total mass of the aerial manipulator is 2.5 kg. We implement the trajectory-tracking controller proposed in [14], as a trajectory tracking controller to generate the desired thrust and angular velocities. In this experiment, for simplicity, we assumed that a single-DoF robotic arm was attached to the aerial robot. Details of the experimental setup are provided in [9].

### *B. EXPERIMENTAL RESULTS*

On the basis of the motion-planning framework discussed in Section III, we performed a experiment with two unknown obstacles, which are shown in Fig. 9. In the experiment, we assumed $v_{max} = 1/3$ m/s and $\kappa_{max} = 3.5$ m. The experimental data are shown in Figs. 10-11. In Fig. 10, the black line indicates the size of an obstacle and the black dotted line is the imaginary size of an obstacle, used for calculating the desired centroid trajectory $\mathbf{x}_c^f$. Figs. 11(a-b) present the trajectory tracking results for a common object. The red and black lines denote the desired and the actual state, respectively. In our experiment, since the size of a hexacopter was relatively large compared with the size of the indoor flight area, a strong downwash effect was produced by the other manipulator. Therefore, the velocity of a manipulator violated the velocity constraint. However, this problem can be easily solved through the appropriate selection of the velocity constraint and by considering a safety factor. Our experiment confirmed that the proposed algorithm shows satisfactory tracking performance in the obstacle environments while satisfying the velocity and curvature constraints.

## VI. CONCLUSION

This paper presents a real-time path planning and smoothing algorithm for cooperative aerial manipulators in obstacle environments; the algorithm satisfies velocity and curvature constraints. For the transportation of an object, the desired centroid trajectory was generated by using a streamline of an HPF and a smooth activation function. To consider the velocity and curvature constraints for reducing the internal force, we proposed a new trajectory generation algorithm that does not use any complex optimization approach. Subsequently, the centroid trajectory was converted to the desired trajectory in joint space by using the formation control method. On the basis of the decentralized dynamics with the aid of a controller, each aerial manipulator follows its own trajectory. We performed successful flight simulation and experiment, by using multiple aerial manipulators. A comparison of the proposed algorithm with the previous method involving DMPs showed that the former requires a smaller control input and generates a smaller internal force.

## ACKNOWLEDGMENT

The authors would like to thank Hoseong Seo and Dohyun Jang for their assistance during experimental validation.

## REFERENCES

[1] W. Li and R. Xiong, "Dynamical obstacle avoidance of task-constrained mobile manipulation using model predictive control," *IEEE Access*, vol. 7, pp. 88301–88311, 2019.

[2] D. Sieber and S. Hirche, "Human-guided multirobot cooperative manipulation," *IEEE Trans. Control Syst. Technol.*, vol. 27, no. 4, pp. 1492–1509, Jul. 2019.

[3] H.-N. Nguyen, S. Park, J. Park, and D. Lee, "A novel robotic platform for aerial manipulation using quadrotors as rotating thrust generators," *IEEE Trans. Robot.*, vol. 34, no. 2, pp. 353–369, Apr. 2018.

[4] N. Michael, J. Fink, and V. Kumar, "Cooperative manipulation and transportation with aerial robots," *Auto. Robot.*, vol. 30, no. 1, pp. 73–86, 2011.

[5] P. O. Pereira, P. Roque, and D. V. Dimarogonas, "Asymmetric collaborative bar stabilization tethered to two heterogeneous aerial vehicles," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 5247–5253.

[6] H. Yang and D. Lee, "Hierarchical cooperative control framework of multiple quadrotor-manipulator systems," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2015, pp. 4656–4662.

[7] J. Alonso-Mora, S. Baker, and D. Rus, "Multi-robot formation control and object transport in dynamic environments via constrained optimization," *Int. J. Robot. Res.*, vol. 36, no. 9, pp. 1000–1021, 2017.

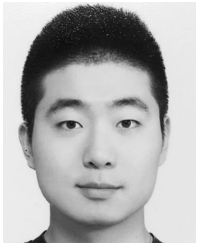
[8] G. Muscio, F. Pierri, M. A. Trujillo, E. Cataldi, G. Antonelli, F. Caccavale, A. Viguria, S. Chiaverini, and A. Ollero, "Coordinated control of aerial robotic manipulators: Theory and experiments," *IEEE Trans. Control Syst. Technol.*, vol. 26, no. 4, pp. 1406–1413, Jul. 2018.

[9] H. Kim, H. Seo, C. Y. Son, H. Lee, S. Kim, and H. J. Kim, "Cooperation in the air: A learning-based approach for the efficient motion planning of aerial manipulators," *IEEE Robot. Autom. Mag.*, vol. 25, no. 4, pp. 76–85, Dec. 2018.

[10] A. Caballero, A. Suarez, F. Real, V. M. Vega, M. Bejar, A. Rodriguez-Castaño, and A. Ollero, "First experimental results on motion planning for transportation in aerial long-reach manipulators with two arms," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 8471–8477.

[11] H. Lee, H. Kim, W. Kim, and H. J. Kim, "An integrated framework for cooperative aerial manipulators in unknown environments," *IEEE Robot. Autom. Lett.*, vol. 3, no. 3, pp. 2307–2314, Jul. 2018.
[12] D. Lau, J. Eden, and D. Oetomo, "Fluid motion planner for nonholonomic 3-D mobile robots with kinematic constraints," *IEEE Trans. Robot.*, vol. 31, no. 6, pp. 1537–1547, Dec. 2015.
[13] Y. Yu and V. Lippiello, "6D pose task trajectory tracking for a class of 3D aerial manipulator from differential flatness," *IEEE Access*, vol. 7, pp. 52257–52265, 2019.
[14] H. Lee, H. Kim, and H. J. Kim, "Planning and control for collision-free cooperative aerial transportation," *IEEE Trans. Autom. Sci. Eng.*, vol. 15, no. 1, pp. 189–201, Jan. 2018.
[15] A. Perez, S. Karaman, A. Shkolnik, E. Frazzoli, S. Teller, and M. R. Walter, "Asymptotically-optimal path planning for manipulation using incremental sampling-based algorithms," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2011, pp. 4307–4313.
[16] R. Conti, E. Meli, A. Ridolfi, and B. Allotta, "An innovative decentralized strategy for I-AUVs cooperative manipulation tasks," *Robot. Auto. syst.*, vol. 72, pp. 261–276, Oct. 2015.
[17] Y. Qi, J. Wang, and J. Shan, "Aerial cooperative transporting and assembling control using multiple quadrotor–manipulator systems," *Int. J. Syst. Sci.*, vol. 49, no. 3, pp. 662–676, 2018.
[18] V. Gazi, "Swarm aggregations using artificial potentials and sliding-mode control," *IEEE Trans. Robot.*, vol. 21, no. 6, pp. 1208–1214, Dec. 2005.
[19] J. Sun, J. Tang, and S. Lao, "Collision avoidance for cooperative UAVs with optimized artificial potential field algorithm," *IEEE Access*, vol. 5, pp. 18382–18390, 2017.
[20] A.-R. Merheb, V. Gazi, and N. Sezer-Uzol, "Implementation studies of robot swarm navigation using potential functions and panel methods," *IEEE/ASME Trans. Mechatronics*, vol. 21, no. 5, pp. 2556–2567, Oct. 2016.
[21] P.-L. Kuo, C.-H. Wang, H.-J. Chou, and J.-S. Liu, "A real-time hydrodynamic-based obstacle avoidance system for non-holonomic mobile robots with curvature constraints," *Appl. Sci.*, vol. 8, no. 11, p. 2144, 2018.
[22] S. Erhart and S. Hirche, "Model and analysis of the interaction dynamics in cooperative manipulation tasks," *IEEE Trans. Robot.*, vol. 32, no. 3, pp. 672–683, Jun. 2016.
[23] H. Lee and H. J. Kim, "Estimation, control, and planning for autonomous aerial transportation," *IEEE Trans. Ind. Electron.*, vol. 64, no. 4, pp. 3369–3379, Apr. 2017.
[24] R. Daily and D. M. Bevly, "Harmonic potential field path planning for high speed vehicles," in *Proc. Amer. Control Conf.*, Jun. 2008, pp. 4609–4614.
[25] S. S. Ge and Y. J. Cui, "New potential functions for mobile robot path planning," *IEEE Trans. Robot. Autom.*, vol. 16, no. 5, pp. 615–620, Oct. 2000.

**HYEONBEOM LEE** received the B.S. degree in mechanical and control engineering from Handong Global University, in 2011, and the M.S. and Ph.D. degrees in mechanical and aerospace engineering from Seoul National University, in 2013 and 2017, respectively. From 2017 to 2018, he was a Senior Researcher with the Korea Institute of Machinery and Materials (KIMM). In September 2018, he joined the Department of Electronics Engineering, Kyungpook National University, Deagu, South Korea, as an Assistant Professor. His research interests include the autonomous navigation of aerial robots and mobile manipulators.



**CLARK YOUNGDONG SON** received the B.S. degree in mechanical engineering from Sungkyunkwan University, in 2015. He is currently pursuing the Ph.D. degree with the Department of Mechanical and Aerospace Engineering, Seoul National University, Seoul, South Korea. His current research interests include aerial robots and motion planning of aerial robots.

**H. JIN KIM** received the B.S. degree from the Korea Advanced Institute of Technology (KAIST), in 1995, and the M.S. and Ph.D. degrees in mechanical engineering from the University of California at Berkeley (UC Berkeley), Berkeley, CA, USA, in 1999 and 2001, respectively. From 2002 to 2004, she was a Postdoctoral Researcher in electrical engineering and computer science (EECS) with UC Berkeley. In September 2004, she joined the Department of Mechanical and Aerospace Engineering, Seoul National University, Seoul, South Korea, as an Assistant Professor, where she is currently a Professor. Her research interests include the intelligent control of robotic systems and motion planning.