Assistive Collision Avoidance for Quadrotor Swarm Teleoperation

Dingjiang Zhou[1] and Mac Schwager[2]

***Abstract*— This paper presents a method for controlling quadrotor swarms in an environment with obstacles, with an intuitive human-swarm interface operated by a single human user. Our method allows for the quadrotor swarm to maintain a desired formation, while also keeping the quadrotors a safe distance from obstacles and from one another. We use a Virtual Rigid Body abstraction to provide a bridge between the single human user and the quadrotor swarm, so that a human user can fly an arbitrarily large quadrotor swarm from a single joystick. By applying multiple vector fields, collisions are automatically avoided within the swarm of quadrotors, and between the quadrotors and obstacles, while the Virtual Rigid Body is controlled by the human user. Our method is demonstrated in hardware experiments with groups of quadrotor micro aerial vehicles teleoperated by a single human operator in a motion capture system.**

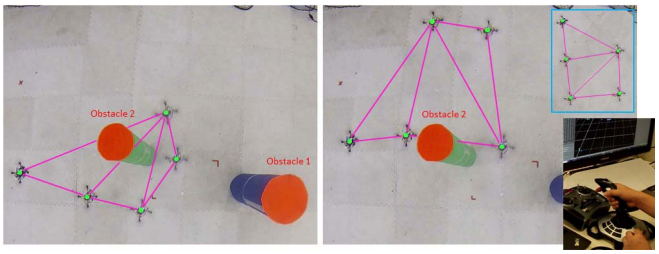


Fig. 1. Two snapshots from our experiment video. A swarm of five KMel Nano Plus quadrotors teleoperated by a gaming joystick in an environment with two obstacles. The "Trapezoid" formation of the swarm of five quadrotors distorts itself using vector fields to avoid collisions with the obstacles. The desired "Trapezoid" formation is shown in the upper-right conner. Our video is available at `http://sites.bu.edu/msl/vrb-obstacles/`.

I. Introduction

In this paper, we propose a teleoperation architecture for a single human user to control an arbitrarily large swarm of quadrotors through a cluttered environment with a standard gaming joystick. The user controls the whole swarm as a single body, while local formation control and collision avoidance is taken care of autonomously by each robot. Our approach can be useful in security, surveillance, and search and rescue applications, in which a human operator must maneuver a swarm through, e.g., a cluttered building, a forest, or a disaster site. By allowing the quadrotors to autonomously avoid collisions, the human user can focus on the overall maneuver of the swam, instead of taking care of each quadrotor to avoid the obstacles in the environment. Based on the abstraction of a *Virtual Rigid Body* proposed in our previous work [1], [2], we implement an intuitive human-swarm interface with a standard gaming joystick to fly a quadrotor swarm from a single human user. Our assistive collision avoidance algorithm supplements the quadrotor swarm control, so that the maneuver of the swarm is not interrupted by the obstacles. Our strategy is integrated with differential flatness-based control techniques, which have been used in the past primarily to design controllers for single quadrotors to execute agile trajectories [3], [4].

Our first idea is to control the group of quadrotors as a single body with a standard gaming joystick, in the framework of Virtual Rigid Body (VRB). The VRB allows us to decouple the trajectory of the whole swarm from the trajectories of the individual quadrotors within the swarm, and allows us to generate the trajectory of the whole swarm from a joystick in real time. As long as the smoothness requirement of the trajectory is satisfied in SE(3), each quadrotor computes its own dynamically feasible SE(3) trajectory based on the differential flatness property of its dynamics. Our second idea is to let the collision avoidance be achieved automatically during the flight such that the human user will not be overwhelmed by controlling each single quadrotor to avoid colliding to the obstacle, and one another. Inspired by our previous work [4], in which we used differential flatness based control together with a vector field method for a single quadrotor to avoid a static obstacle, here we apply multiple vector fields for the obstacles, and also for the quadrotors, in order to keep a desired formation, as well as to avoid collision with the obstacles.

We demonstrate the effectiveness of our strategy in two sets of hardware experiments with KMel Nano Plus quadrotors. One experiment is to control a swarm of five quadrotors in a "Trapezoid" formation in an environment with two obstacles, and the another is to control the swarm in a "Line" formation in the same environment. In both experiments, the quadrotors are controlled by a Logitech Extreme 3D Pro joystick operated by the first author of this paper. Figure 1 shows two snapshots of our experiment.

There exists an extensive literature in formation control. For example, [5], [6] proposed a formation control strategy in which a leader quadrotor flies an agile trajectory while other follower quadrotors control themselves to keep a desired formation with respect to the leader. In [7] the authors achieved formation control for a team of quadrotors using only onboard vision, without an external positioning system. In [8], control and trajectory generation tools for controlling swarms of micro quadrotors between different formations were described. Also in [9], the authors displayed 3D features with multiple aerial vehicles and shown animation with

This work was supported in part by NSF grants IIS-1350904 and CNS-1330008, and ONR grant N00014-12-1-1000. We are grateful for this support.

[1]D. Zhou is with the Department of Mechanical Engineering, Boston University, Boston, MA 02215, USA `zdj@bu.edu`

[2]M. Schwager is with the Department of Aeronautics and Astronautics, Stanford University, Stanford, CA 94305, USA `schwager@stanford.edu`

a sequence of 3D features. The teleoperation of mobile robot swarm is considered in [10], [11], where multiple nonholonomic vehicles were controlled in a leader-follower formation and the leader robot was remotely driven to a desired velocity. In [12], the authors used a haptic interface to control the formation of a group of quadrotors. Gesture based human-swarm interaction is seen in [13], in which authors deciphered human gestures to drive multiple ground mobile robots and let the robots show different faces. Obstacle avoidance for single quadrotor and multiple quadrotors is also heavily researched in the literature. For a single quadrotor, control through a hatpic interface control is demonstrated in [14] where the pilot received force feedback when obstacles are close, and in [15] the authors proposed an approach to construct a novel dynamic kinesthetic boundary (DKB) to aid a pilot to navigate an aerial robotic vehicle through a cluttered environment. In [16], the authors provided a method to assist the operator to fly a quadrotor to perform collision avoidance by predicting its future trajectory. The paper [17] presents a method to assist the teleoperation of a quadrotor by constructing the nearby environment using sonar sensors. For multi-quadrotor scenarios, the authors of [18] formulated a mixed-integer linear program (MILP) approach to generate optimal trajectories for the aircrafts to avoid collisions, and in [19], mixed-integer quadratic programming (MIQP) method was proposed to generate optimal trajectories for teams of heterogeneous quadrotors in 3D environment with obstacles.

Our work is different from the above works in two aspects. Firstly, we apply our Virtual Rigid Body abstraction such that the human-swarm interface is naturally integrated in our control. Secondly, our assistive collision avoidance algorithm is applied to the swarm instead of a single quadrotor, and it is implemented experimentally in real time. This makes it directly applicable to a real-world setting (assuming the quadrotors have an adequate perception of their surroundings).

The remainder of this paper is organized as follows. The quadrotor dynamics and our Virtual Rigid Body abstraction, are presented in Section II. In Section III, we introduce our intuitive human-swarm interface which is used to control a quadrotor swarm by a single joystick. Our vector field based assistive collision avoidance algorithm is then introduced in Section IV. In Section V, we present our experiments with KMel Nano Plus quadrotors in an environment with two obstacles, controlled from a standard gaming joystick by a human user. Finally, we offer our concluding remarks in Section VI.

II. Preliminary

In this section, we give a necessary background on quadrotor dynamics, and our Virtual Rigid Body abstraction.

A. Quadrotor Dynamics

A quadrotor is well-modeled as a rigid body with forces and torques applied from the four rotors and gravity [3]. The relevant forces, moments, and coordinate frames are shown in Figure 2.

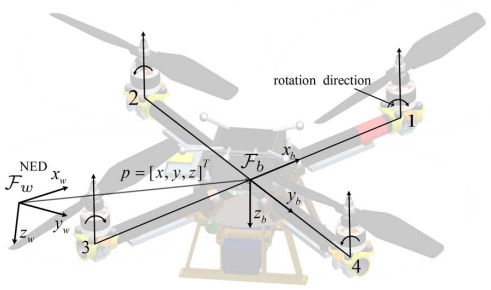


Fig. 2. Coordinate frames of a quadrotor. The North-East-Down (NED) global reference frame is denoted by $\mathcal{F}_w$, and the quadrotor body-fixed frame is denoted by $\mathcal{F}_b$.

We define a global reference frame $\mathcal{F}_w$ with a North-East-Down system, and have a body-fixed frame $\mathcal{F}_b$ attached to the quadrotor, with its origin at the center of mass. The orientation of the quadrotor is defined by the rotation matrix $\mathbf{R}$, parameterized with ZYX Euler angles, i.e., roll ϕ, pitch θ and yaw ψ [1], which are also known as TaitBryan angles. The quadrotor dynamics are given by the nonlinear system of equations

$$\begin{cases} \dot{\mathbf{v}} = \mathbf{g} + \dfrac{1}{m}\mathbf{R}\mathbf{f} & (1) \\ \dot{\mathbf{R}} = \mathbf{R}\boldsymbol{\Omega} & (2) \\ \dot{\boldsymbol{\omega}}_b = \mathbf{J}^{-1}\boldsymbol{\tau} - \mathbf{J}^{-1}\boldsymbol{\Omega}\mathbf{J}\boldsymbol{\omega}_b & (3) \\ \dot{\mathbf{p}} = \mathbf{v}, & (4) \end{cases}$$

where $\mathbf{v} = [v_x, v_y, v_z]^T$ is the velocity in the global frame $\mathcal{F}_w$, $\mathbf{g} = [0, 0, g]^T$ is the acceleration due to gravity, m is the mass, and $\mathbf{f}$ is the total thrust force generated from the rotors, with $\mathbf{f} = [0, 0, f_z]^T$, where f_z is the summation of the four rotors' thrust, hence $\mathbf{f}$ is aligned with the negative vertical body-fixed direction $-\mathbf{z}_b$ of the body frame $\mathcal{F}_b$. The rotation matrix $\mathbf{R}$ is defined with Euler angles ϕ, θ and ψ. The angular velocity of the quadrotor, $\boldsymbol{\omega}_b = [\omega_x, \omega_y, \omega_z]^T$, is expressed in the body frame $\mathcal{F}_b$, and $\boldsymbol{\Omega} = \boldsymbol{\omega}_b^\wedge = [0, -\omega_z, \omega_y; \omega_z, 0, -\omega_x; -\omega_y, \omega_x, 0]$ is the tensor form of $\boldsymbol{\omega}_b$. The torque generated from the rotors on the quadrotor is given by $\boldsymbol{\tau} = [\tau_x, \tau_y, \tau_z]^T$, expressed in the body frame $\mathcal{F}_b$. The inertia matrix of the quadrotor is $\mathbf{J}$, and $\mathbf{p} = [x, y, z]^T$ is the position of the quadrotor in the global frame $\mathcal{F}_w$. The quadrotor's inputs are the total thrust and the three torques, $\boldsymbol{\mu} = [f_z, \tau_x, \tau_y, \tau_z]^T$, which is in four dimensions, and the system has a 12 dimensional state, i.e., $\boldsymbol{\xi} = [x, y, z, v_x, v_y, v_z, \psi, \theta, \phi, \omega_x, \omega_y, \omega_z]^T$.

Planning a trajectory to be executed by a quadrotor is not trivial, as one must find a trajectory that satisfies the dynamics (1)–(4), and find the control input that produces that trajectory. We call a trajectory that satisfies the dynamics of a quadrotor for some input signal dynamically feasible, which is defined formally as follows.

Definition 1: (Dynamically Feasible Trajectory) A quadrotor trajectory $\boldsymbol{\xi}(t)$ is called dynamically feasible over a time interval $[t_1, t_2]$ if there exists a control input $\boldsymbol{\mu}(t)$ such that $\boldsymbol{\xi}(t)$ and $\boldsymbol{\mu}(t)$ satisfy the equations (1)–(4) for all $t_1 \leq t \leq t_2$.

Differential flatness [20] is a property of some nonlinear control systems that greatly simplifies the problem of finding dynamically feasible trajectories. If a control system is differentially flat, its states and inputs can be written in terms of a smaller number of, so called, flat outputs, and some number of time derivatives of those outputs [21]. Quadrotor dynamics are known to be differentially flat [1], [3]. This is useful in trajectory planning because one can plan an arbitrary trajectory for the flat outputs (as long as it is sufficiently smooth), and analytically find a dynamically feasible quadrotor trajectory, and the control inputs required to execute that trajectory. The key tool from differential flatness is called the *endogenous transformation* [20]. The endogenous transformation for quadrotor dynamics can be found in previous work [1], [3], which maps the flat outputs $\boldsymbol{\sigma} = [x, y, z, \psi]^T$ of a quadrotor to its 12 dimensional states $\boldsymbol{\xi}$ and 4 dimensional control inputs $\boldsymbol{\mu}$.

B. Virtual Rigid Body

In our previous work [1], [2], we defined the concept of Virtual Rigid Body, which is a key idea in our approach to control quadrotor swarms.

Consider a swarm of N robots labeled by $\{1, 2, \ldots, N\}$. Let $\mathcal{F}_i$ denote the local reference frame of robot i, and we denote by $\mathbf{p}_i(t) \in \mathbb{R}^3$ the position, and $\mathbf{R}_i(t) \in SO(3)$ the orientation of robot i with respect to the global frame $\mathcal{F}_w$, at time t.

Definition 2 (Virtual Rigid Body): A Virtual Rigid Body (VRB) is a group of N robots and a local reference frame $\mathcal{F}_v$, in which the local positions of the robots are specified by a set of potentially time-varying vectors $\{\mathbf{r}_1(t), \mathbf{r}_2(t), \cdots, \mathbf{r}_N(t)\}$.

An example of a Virtual Rigid Body of three robots is shown in Figure 3, in which the *formation* and *transformation* are defined as follows.

Definition 3 (Formation): A formation $\boldsymbol{\Pi}$ is a Virtual Rigid Body with constant local positions $\{\mathbf{r}_1, \mathbf{r}_2, \cdots, \mathbf{r}_N\}$ in $\mathcal{F}_v$ for a group of N robots associated with a time period $T_{\boldsymbol{\Pi}} > 0$.

Definition 4 (Transformation): A transformation $\boldsymbol{\Phi}$ is a Virtual Rigid Body with time varying local positions $\{\mathbf{r}_1(t), \mathbf{r}_2(t), \cdots, \mathbf{r}_N(t)\}$ in $\mathcal{F}_v$ for a group of N robots associated with a time period $T_{\boldsymbol{\Phi}} > 0$.

Let $\mathbf{p}_v(t) \in \mathbb{R}^3$ and $\mathbf{R}_v(t) \in SO(3)$ denote the position and orientation of the origin of $\mathcal{F}_v$ in the global reference frame $\mathcal{F}_w$ at time t, respectively. Since robots' positions are defined locally in the Virtual Rigid Body as in Definition 2, the relationship between $\mathbf{p}_i(t)$ and $\mathbf{r}_i(t)$ for robot i is described as

$$\mathbf{p}_i = \mathbf{p}_v + \mathbf{R}_v \mathbf{r}_i, \quad i = 1, 2, \cdots, N, \tag{5}$$

where "(t)" is dropped for simplicity.

We define the trajectory of a Virtual Rigid Body to be the position and orientation of its local frame $\mathcal{F}_v$ as a function of time,

$$\boldsymbol{\delta}_v(t) = (\mathbf{p}_v(t), \mathbf{R}_v(t)) \in \mathcal{C}^4 : \ \mathbb{R}_{\geq 0} \mapsto \mathbb{R}^3 \times SO(3). \tag{6}$$

An example of the Virtual Rigid Body trajectory $\boldsymbol{\delta}_v(t)$ is shown again in Figure 3, in which it is depicted as a black dashed line. In correspondence, the trajectory of robot i in the global frame is denoted by $\boldsymbol{\delta}_i(t) = (\mathbf{p}_i(t), \mathbf{R}_i(t)) \in \mathcal{C}^4$. In Figure 3, the trajectories for the three robots are depicted in red, blue and green dashed lines, respectively.

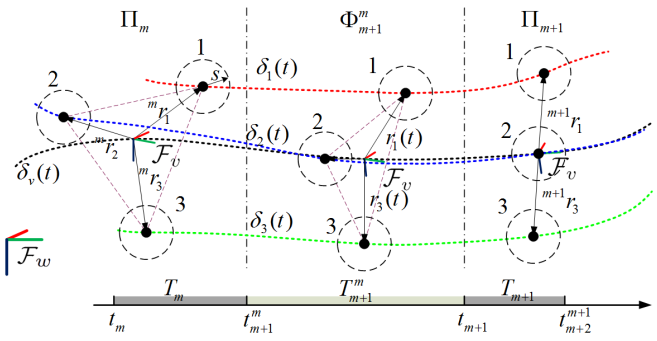


Fig. 3. The concept of a Virtual Rigid Body with a group of three robots. The VRB is in a formation $\boldsymbol{\Pi}_m$ when $t_m < t \leq t^m_{m+1}$, and $\boldsymbol{\Pi}_{m+1}$ when $t_{m+1} < t \leq t^{m+1}_{m+2}$, and it is in a transformation $\boldsymbol{\Phi}^m_{m+1}$ when $t^m_{m+1} \leq t \leq t_{m+1}$.

III. Intuitive Human Interface

Here we describe the human-swarm interface component of our control architecture. With our human-swarm interface, the human operator is able to control an arbitrarily large quadrotor swarm as if the swarm were a single aircraft, without be overwhelmed by the excessive number of degrees of freedom of the quadrotor swarm. Our Virtual Rigid Body, which abstracts the quadrotor swarm as a single body, decouples the trajectory generation for a group of robots into two subproblems, (i) generating the trajectory of the Virtual Rigid Body in the global reference frame, and (ii) generating the trajectory for each individual quadrotor within the VRB local reference frame. Our human-swarm interface hence solves the first subproblem, by means of generating a trajectory for the Virtual Rigid Body from a standard joystick. The second subproblem, however, is solved with a formation library, which stores a set of formations, as well as transformations between any pair of formations, for the Virtual Rigid Body. To fully appreciate the implementation of a formation library, please refer to our previous work [2].

The joystick gives commands to our base computer, which interprets the commands as a state trajectory for the Virtual Rigid Body, then the commanded VRB state is broadcast to the quadrotors, as shown in Figure 4. This control method is intuitive, since the joystick is widely used in flight simulators and computer games. The axes of the joystick are interpreted as commanded Euler angles and spatial velocity for the VRB, while the buttons are used to select different formations from the formation library. The goal of this section is to describe the map between the joystick signal, namely, Euler angles and thrust $(\phi_J, \theta_J, \psi_J, f_J)$ and their first derivatives $(\dot{\phi}_J, \dot{\theta}_J, \dot{\psi}_J, \dot{f}_J)$, to the Virtual Rigid Body trajectory $\boldsymbol{\delta}_v(t)$ in real time.

The raw data from the joystick is noisy due to unavoidably jerky motion from the human hand. To smooth this signal, we apply a first order filter. Notice that the roll ϕ_J, pitch θ_J and yaw rate $\dot{\psi}_J$ are from the joystick directly, so that the

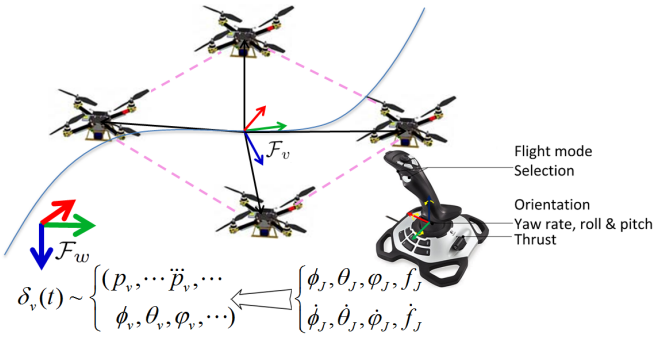


Fig. 4. The concept of controlling a quadrotor swarm from a standard joystick. The 4 axes on the joystick are assigned to the three Euler angles and the thrust, and a few among the 12 buttons are assigned to different formations.

roll rate $\dot{\phi}_J$ and the pitch rate $\dot{\theta}_J$ are obtained by numerical differentiation from ϕ_J and θ_J, while the yaw ψ_J is obtained by numerical integration from $\dot{\psi}_J$. Also, $\ddot{\psi}_J$ is obtained by numerical differentiation from $\dot{\psi}_J$. Given these filtered joystick commands, we must find a VRB trajectory that the user intends.

We impose virtual dynamics on the Virtual Rigid Body to map the joystick commands to a VRB trajectory. The VRB dynamics can be as simple as possible, since there is no inherent physical dynamic constraints on the VRB. In this paper, we let the VRB have single integrator dynamics,

$$\dot{\mathbf{x}}_v = \boldsymbol{\upsilon}, \tag{7}$$

where the state is $\mathbf{x}_v = [\mathbf{p}_v^T, \psi_v]^T$. Note that roll ϕ_v and pitch θ_v of the VRB are not modeled since we set them to zero. With this model, the VRB trajectory $\boldsymbol{\delta}_v(t)$ defined in (6) is simplified as the position trajectory $\mathbf{p}_v(t)$ with the yaw trajectory $\psi_v(t)$.

We apply again a first order filter to the input $\boldsymbol{\upsilon}$ in (7),

$$\dot{\boldsymbol{\upsilon}} = -a\boldsymbol{\upsilon} + \mathbf{u}_1, \tag{8}$$

where a is a positive parameter, and the input $\mathbf{u}_1$ is from the joystick by

$$\mathbf{u}_1 = \begin{bmatrix} e_1\theta_J \\ e_1\phi_J \\ e_2 f_J \\ e_3\dot{\psi}_J \end{bmatrix}, \tag{9}$$

in real time, where e_1, e_2 and e_3 are positive constant scaling factors, such that e_1 and e_2 scale the spatial speed, and e_3 scales the yaw rate of the Virtual Rigid Body. The three scaling factors, along with the positive parameter a in (8), can be used as four variables for tuning the aggressiveness of the maneuver. These parameters should be tuned so as to avoid actuator saturation.

From (7) to (9), we obtain the VRB position trajectory $\mathbf{p}_v(t)$, yaw angle trajectory $\psi_v(t)$, as well as its velocity $\mathbf{v}_v(t)$. However, we require that the Virtual Rigid Body to be four times continuously differentiable (in $\mathcal{C}^4$) in order to use differential flatness based control tools. Therefore, the VRB acceleration $\dot{\mathbf{v}}_v(t) \equiv \ddot{\mathbf{p}}_v(t)$, jerk $\ddot{\mathbf{v}}_v(t) \equiv \dddot{\mathbf{p}}_v(t)$, and snap $\dddot{\mathbf{v}}_v(t) \equiv \ddddot{\mathbf{p}}_v(t)$ must be found from the joystick commands. Similar to (8), we apply a first order filter to the acceleration of the VRB,

$$\dot{\mathbf{a}}_v = -a\mathbf{a}_v + \mathbf{u}_2, \tag{10}$$

where the input $\mathbf{u}_2$ are from the joystick by

$$\mathbf{u}_2 = \begin{bmatrix} e_1\dot{\theta}_J \\ e_1\dot{\phi}_J \\ e_2\dot{f}_J \end{bmatrix}.$$

Finally, we obtain jerk and snap directly by

$$\begin{cases} \mathbf{j}_v = \dot{\mathbf{a}}_v \\ \mathbf{s}_v = \dot{\mathbf{j}}_v \end{cases}. \tag{11}$$

In conclusion, (7)–(11) fully define the trajectory $\boldsymbol{\delta}_v(t)$ for the Virtual Rigid Body. With the VRB trajectory $\boldsymbol{\delta}_v(t)$, and the formation or transformation of the VRB from a formation library, one can apply the methods in our previous work [1] to find the trajectory for each quadrotor.

IV. Assistive Collision Avoidance

In this section, we introduce the collision avoidance component of our control architecture, which supplements the previous human-swarm interface to assist the human user by autonomously avoiding collisions with obstacles. Using multiple interacting vector fields, which is similar to a potential field, the algorithm allows for the formation to deform enough to avoid collisions with obstacles, and relax back into the formation when far away from obstacles. This behavior can be seen in our simulation in Section IV-D and our experiments in Section V. One possible draw back of potential field-based methods is that the deadlocks may occur [10], [11]. However, since the swarm is teleoperated by a human user in our case, this can be less of a problem. The user can see when a deadlock is occurring, and reactively maneuver the swarm to dislodge the quadrotors that are stuck in a local minimum.

Our algorithm for collision avoidance is applied to environments like the scenario shown in Figure 5, in which a swarm of five quadrotors is under control from a human user in an environment with two obstacles. The obstacles are modeled as cylinders with different radii. Since we focus on formation control and collision avoidance, we assume that the robots sense their local environment precisely.

In our strategy, the obstacles affect the maneuver of the quadrotor swarm in two different ways. First, the obstacles influence the commanded velocity $\mathbf{v}_v$, as well as the commanded acceleration $\mathbf{a}_v$ of the VRB through a vector field, such that the VRB will slow down or change its heading direction when there is an obstacle in its path. Second, the obstacles influence the individual quadrotors that are close to it, also through vector fields, that each individual quadrotor takes its own action to deviate from its original heading direction in order to avoid collision to the obstacles. Additionally, to avoid inter-vehicle collision, each quadrotor considers all the other quadrotors as dynamical obstacles with repelling vector fields. Meanwhile, the desired formation is recovered after the swarm passes through the obstacle area by formation control algorithm.

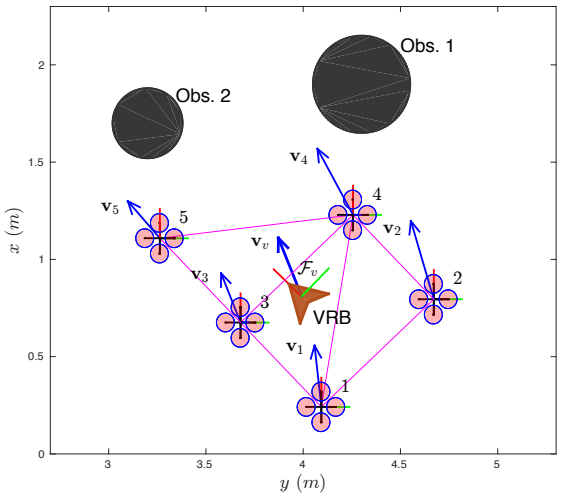


Fig. 5. Top view of a scenario with two obstacles in the environment. The obstacles are plotted as solid black disks. The magenta lines show the connectivity of the quadrotor swarm. The orange arrowhead shape in the middle denotes the VRB of our quadrotor swarm, and the $\mathbf{v}_v$ depicted by a thick blue arrow is the commanded velocity of the VRB from the joystick. The velocity of all quadrotors are denoted as $\mathbf{v}_1$ to $\mathbf{v}_5$, respectively. The velocities are different from the VRB velocity $\mathbf{v}_v$, since the VRB is in rotation about the z axis of $\mathcal{F}_v$.

Our control algorithm for collision avoidance runs in real time and is distributed among the quadrotor swarm, with the exception of the component that acts on the VRB, as described below. For the scenario as shown in Figure 5, we apply the vector field method from our previous work [4], in which it was applied for a single quadrotor to avoid a static "virtual" obstacle in the environment. With running this control algorithm, the dynamical feasible trajectory of each quadrotor described in Definition 1, and the smoothness requirement of the states and inputs, can be satisfied. A system diagram of our control architecture is shown in Figure 6.

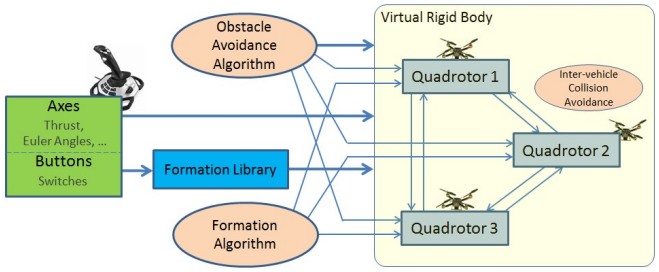


Fig. 6. System diagram of controlling a quadrotor swarm with collision avoidance. The human user controls the maneuver of the Virtual Rigid Body with a joystick, and selects a desired formation from the formation library. The obstacle avoidance algorithm affects the VRB, and as well as each individual quadrotor in the VRB. The interaction between each pair of neighboring quadrotors avoids inter-vehicle collision, and the formation algorithm keeps the desired formation for the quadrotor swarm.

A. Formation Vector Fields

In our Virtual Rigid Body framework, we apply a vector field for each quadrotor in the VRB local frame $\mathcal{F}_v$, such that when the quadrotor deviates from its desired position, a forcing vector acts on the quadrotor to pull it back to the desired position in the formation. Here we take advantage of our VRB abstraction, so that instead of considering the maneuver of the quadrotors in the global frame, we only consider their maneuver in the VRB local frame.

Let the desired position of quadrotor i in the VRB local frame $\mathcal{F}_v$ be denoted as $\mathbf{r}_i^d$, then the vector in the vector field at position $\mathbf{r}$ for quadrotor i is defined as

$$\mathbf{V}_{0,i}(\mathbf{r}) = A(\mathbf{r}_i^d - \mathbf{r}), \quad i = 1, 2, \cdots, N, \tag{12}$$

where A is a positive constant parameter. As long as the quadrotor can follow its vector field, it will eventually converge to its desired position in the VRB local frame, when the environment is obstacle free. On the other hand, when the desired formation is feasible, this vector field will also keep that desired formation.

An example of a 2D version of this vector field for each quadrotor is shown in Figure 7, in which different color denotes the vector field for different quadrotor. Note that all five vector fields actually act on the quadrotor at once. Another example of this vector field for quadrotor 1 is shown in Figure 9, plotted in gray arrows.

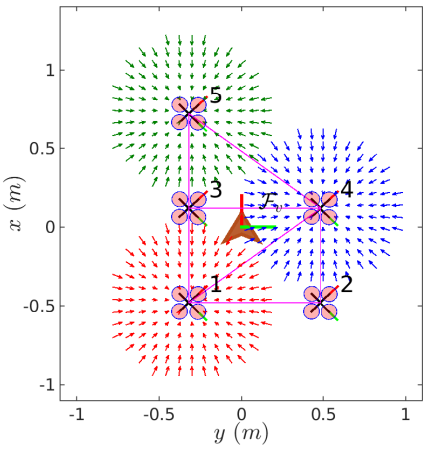
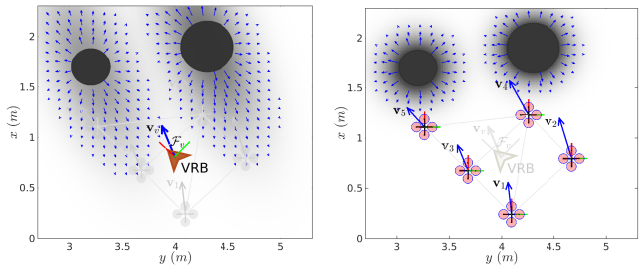


Fig. 7. Formation vector fields for quadrotor 1, 4 and 5 in the VRB local frame $\mathcal{F}_v$.

B. Vector Fields from Obstacles

As illustrated in Figure 6, our collision avoidance algorithm works at two levels: The first level is that we generate a vector field to force the Virtual Rigid Body to change its commanded velocity $\mathbf{v}_v$ to decrease the aggressiveness of the VRB, and the second level is that we generate another vector field for each obstacle to guarantee collision avoidance for each quadrotor, while the VRB is still tracking its trajectory. The two levels can be unequally responsible in achieving collision avoidance. For example, one can increase the influence of the first level and hence decrease that of the second level, such that the entire swarm can be far away from the obstacles while keeping a desired formation. However, we choose to let the second level to be more responsible for collision avoidance. By doing so, collision avoidance is guaranteed for the quadrotor swarm, while maintaining the desired formation is a secondary priority.

In this paper, for any obstacle, we consider a bounding cylinder that encompasses the obstacle in order to generate a repelling vector field. Figure 8 shows a plot of our repelling vector field for the obstacles in the scenario of Figure 5.



(a) Vector field for the VRB. (b) Vector field for the quadrotors.

Fig. 8. The two vector fields for the VRB and quadrotors, respectively. Each vector field is generated from (13), or (17), with different parameters. The darker color means larger magnitude and the lighter color means smaller magnitude. Intuitively, the VRB, or each single quadrotor, will be pushed toward the light region and avoid the dark region.

1) Vector Field for the Virtual Rigid Body: We assume that the quadrotor swarm maneuvers in the horizontal plane only, so that climbing to a higher altitude to avoid the obstacles is not an option in our setting. For the Virtual Rigid Body, and for a single obstacle, we define the repelling vector from the obstacle at horizontal coordinate $\bar{\mathbf{p}}$ to be a vector pointing in the direction from the obstacle to $\bar{\mathbf{p}}$, with its magnitude to be a Gaussian-like function related to the position and radius of this obstacle, as well as the VRB heading direction. Assuming that there are m obstacles in the environment, and we denote the horizontal position of obstacle k in the global frame $\mathcal{F}_w$ as $\bar{\mathbf{o}}_{w,k}$, then the overall repelling vector in the vector fields generated from the m obstacles at horizontal coordinate $\bar{\mathbf{p}}$ is

$$\mathbf{V}_1(\bar{\mathbf{p}}) = \sum_{k=1}^{m} f_k(\bar{\mathbf{o}}_{w,k}, r_k, \hat{\mathbf{v}}) \frac{\bar{\mathbf{p}} - \bar{\mathbf{o}}_{w,k}}{\|\bar{\mathbf{p}} - \bar{\mathbf{o}}_{w,k}\|}, \tag{13}$$

where the magnitude $f_k(\bar{\mathbf{o}}_{w,k}, r_k, \hat{\mathbf{v}})$ is a function of the obstacle position $\bar{\mathbf{o}}_{w,k}$, radius r_k and the VRB heading direction $\hat{\mathbf{v}} = \mathbf{v}_v / \|\mathbf{v}_v\|$, described as

$$f_k(\bar{\mathbf{o}}_{w,k}, r_k, \hat{\mathbf{v}}) = B_k \exp\Big(- (\bar{\mathbf{p}} - \bar{\mathbf{o}}_{w,k})^T \mathbf{\Sigma}^{-1} (\bar{\mathbf{p}} - \bar{\mathbf{o}}_{w,k}) \Big), \tag{14}$$

where B_k is a positive scalar parameter related to the radius r_k of obstacle k. The value of B_k can be chosen so that the commanded velocity $\mathbf{v}_v$ for the VRB can still overcome the largest repelling vector, since the VRB is virtual. The 2×2 matrix $\mathbf{\Sigma}$ is positive definite and defines the major and minor axes of the dynamic Gaussian-like function by

$$\mathbf{\Sigma}^{-1} = \big[\hat{\mathbf{v}}, \hat{\mathbf{v}}^{\perp}\big] \begin{bmatrix} \frac{1}{2\sigma_1^2} & 0 \\ 0 & \frac{1}{2\sigma_2^2} \end{bmatrix} \big[\hat{\mathbf{v}}, \hat{\mathbf{v}}^{\perp}\big]^T, \tag{15}$$

where $\hat{\mathbf{v}}^{\perp}$ is the perpendicular unit vector of $\hat{\mathbf{v}}$, and σ_1 and σ_2 are the covariance variables. From (15) and from Figure 8(a), one can see that the VRB is repelled more from the obstacle if the VRB is heading towards the obstacle directly, than if it is passing by the obstacles from the side.

The vector field $\mathbf{V}_1$ is shown in Figure 8(a).

2) Vector Field for Quadrotors: In contrast to the vector field for the VRB, we need to design a stronger vector field for the individual quadrotors, such that when a quadrotor gets too close to an obstacle, the repelling vector is powerful enough to "push" the quadrotor away from the obstacle,

$$\mathbf{V}_2(\bar{\mathbf{p}}) = \sum_{k=1}^{m} \Bigg(C_k \exp\Big(- \frac{1}{2\sigma_3^2} (\bar{\mathbf{p}} - \bar{\mathbf{o}}_{w,k})^T (\bar{\mathbf{p}} - \bar{\mathbf{o}}_{w,k}) \Big) \cdot \frac{\bar{\mathbf{p}} - \bar{\mathbf{o}}_{w,k}}{\|\bar{\mathbf{p}} - \bar{\mathbf{o}}_{w,k}\|} \Bigg) \tag{16}$$

$$= \sum_{k=1}^{m} \Bigg(C_k \exp\Big(- \frac{1}{2\sigma_3^2} \|\bar{\mathbf{p}} - \bar{\mathbf{o}}_{w,k}\|^2 \Big) \Bigg) \frac{\bar{\mathbf{p}} - \bar{\mathbf{o}}_{w,k}}{\|\bar{\mathbf{p}} - \bar{\mathbf{o}}_{w,k}\|}, \tag{17}$$

where C_k is again a positive scalar parameter related to the radius of obstacle k. The value of C_k must be chosen large enough to guarantee collision avoidance.

The vector field $\mathbf{V}_2$ is shown in Figure 8(b). Equation (16) and (17) are expressed in the global frame, while in our simulation of collision avoidance in Section IV-D, we express them in the VRB local frame by

$$\mathbf{V}_2(\bar{\mathbf{r}}) = \sum_{k=1}^{m} \Bigg(C_k \exp\Big(- \frac{1}{2\sigma_3^2} \|\bar{\mathbf{r}} - \bar{\mathbf{o}}_{v,k}\|^2 \Big) \Bigg) \frac{\bar{\mathbf{r}} - \bar{\mathbf{o}}_{v,k}}{\|\bar{\mathbf{r}} - \bar{\mathbf{o}}_{v,k}\|}, \tag{18}$$

where $\bar{\mathbf{r}}$ denotes the horizontal coordinate in $\mathcal{F}_v$, and $\bar{\mathbf{o}}_{v,k}$ is the horizontal position of obstacle k in $\mathcal{F}_v$. Equation (18) is obvious since $\|\bar{\mathbf{p}} - \bar{\mathbf{o}}_{w,k}\| = \|\bar{\mathbf{r}} - \bar{\mathbf{o}}_{v,k}\|$.

The magnitude of the vector $\mathbf{V}_2$ in the vector field of obstacle 1 in the VRB local frame can be seen in Figure 9.

C. Vector Fields from Quadrotors

Here we also consider inter-vehicle collision avoidance. With the two vector fields described in Section IV-B, we guarantee that the obstacle avoidance is achieved, as seen from our experiment in Section V. To ensure inter-vehicle collision avoidance, we again apply the vector field method to each quadrotor in the VRB local frame $\mathcal{F}_v$. Similar to (18), we have

$$\mathbf{V}_{3,i}(\mathbf{r}) = \sum_{j \in \mathcal{N}_i} \Bigg(D \exp\Big(- \frac{1}{2\sigma_4^2} \|\mathbf{r} - \mathbf{r}_j\|^2 \Big) \Bigg) \frac{\mathbf{r} - \mathbf{r}_j}{\|\mathbf{r} - \mathbf{r}_j\|}, \tag{19}$$

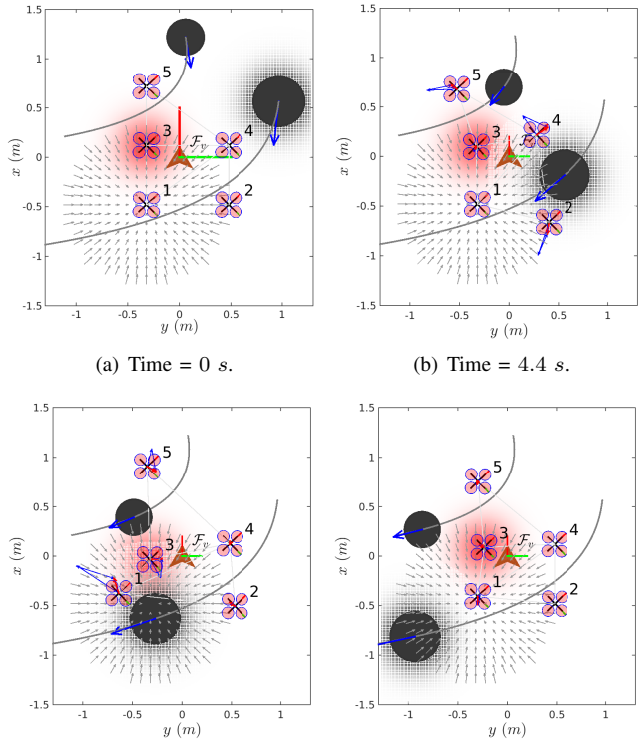
where $i = 1, 2, \cdots, N$, and $\mathcal{N}_i$ denotes the neighbors of quadrotor i. The scale parameter D and covariance σ_4 can be tuned so that the vector field from each quadrotor will not effect the other quadrotors when the swarm is in a desired formation. The magnitude of the vectors of a 2D version of this vector field for quadrotor 3 can be seen in Figure 9.

D. Collision Avoidance Simulation in VRB Local Frame

In previous sections, we defined vector fields to prevent collisions with the obstacles, and with other quadrotors, in either the global frame $\mathcal{F}_w$, or the VRB local frame $\mathcal{F}_v$. In this section, we show the effectiveness of our vector fields in a simulation of a quadrotor swarm with collision avoidance.

By taking the advantage of our Virtual Rigid Body abstraction, instead of considering the static or dynamic obstacles in the global reference frame $\mathcal{F}_w$ in which the quadrotors are maneuvering, we consider dynamic obstacles in the VRB local frame. The global maneuver of each quadrotor is fused from the local maneuver of each quadrotor in $\mathcal{F}_v$ and the global maneuver of the Virtual Rigid Body in $\mathcal{F}_w$ [1]. When collision avoidance is achieved in the VRB local frame, it is automatically achieved in the global frame for the quadrotor swarm.

Figure 9 shows a sequence of snapshots in our simulation starting from the scenario shown in Figure 5, in the VRB local frame $\mathcal{F}_v$. Because the scenario is shown in the VRB local frame, only the obstacles and quadrotors appear to move, while the VRB center appears to be fixed. For this simulation, we pre-planned the trajectories for the obstacles, as plotted by gray lines. These trajectories are designed to be smooth, although they can be more complicated in the real-world.



(a) Time = 0 s. (b) Time = 4.4 s.

(c) Time = 9.0 s. (d) Time = 11.8 s.

Fig. 9. A sequence of snapshots in a simulation of collision avoidance for a quadrotor swarm, shown in the VRB local frame $\mathcal{F}_v$. The velocities of the obstacles are plotted with thick blue arrows, and the velocities of the quadrotors are plotted with thin blue and thick red arrows (the red arrow is a summation of the thin blue arrows). The vector field generated from obstacle 1 is indicated by a gray shadow, and the vector field from quadrotor 3 is indicated by a red shadow. Both vector fields move in $\mathcal{F}_v$ with obstacle 1 and quadrotor 3, respectively. The formation vector field for quadrotor 1 is plotted with gray arrows, and it is static in $\mathcal{F}_v$, since its purpose is to "pull" quadrotor 1 to a desired position. The figures are available from our simulation video at `http://sites.bu.edu/msl/vrb-obstacles/`.

V. Experiments

Our assistive collision avoidance algorithm for the teleoperated quadrotor swarm is here validated experimentally with a group of KMel Nano Plus quadrotors in an environment with two obstacles, as shown in Figure 1. The radii of the obstacles are $r_1 = 0.15m$ and $r_2 = 0.11m$, respectively. The parameters for the vector fields described in Section IV are listed in Table I. These parameters are obtained from tuning in the simulation and experiments.

TABLE I
Vector Fields Parameters for Experiments

	Value	Description.
A	0.70	Magnitude of formation vector field to each quadrotor.
B_1	0.018	Magnitude of vector field of obstacle 1 to the VRB.
B_2	0.0132	Magnitude of vector field of obstacle 2 to the VRB.
σ_1	0.50	Covariance of obstacle vector fields to the VRB.
σ_2	0.25	Covariance of obstacle vector fields to the VRB.
C_1	6.00	Magnitude of vector field of obstacle 1 to quadrotors.
C_2	4.40	Magnitude of vector field of obstacle 2 to quadrotors.
σ_3	0.25	Covariance of obstacle vector fields to quadrotors.
D	1.20	Magnitude of vector fields from quadrotors.
σ_4	0.14	Covariance of vector fields from quadrotors.

In the experiment, position and orientation for the quadrotors are obtained with an OptiTrack motion capture system running at 120Hz. Our experiments share the same code base with our previous work [1], [2], [4] to fly the quadrotors. Once again, a video of our experiments is available at `http://sites.bu.edu/msl/vrb-obstacles/`.

A. Experiment I: Trapezoid Formation

Our first experiment is to control a swarm of five quadrotors flying a "Trapezoid" formation in the experimental arena with the two cylinder obstacles. The "Trapezoid" formation is designed in our formation library. The human user can choose the formation by pressing a button on the joystick. A six-frame sequence of this experiment is shown in Figure 10, but of course, the motion of the quadrotors is difficult to appreciate from the still frames, please refer to our experiment video.

B. Experiment II: Line Formation

The second experiment is the same with experiment I, except that the VRB has a transformation from a "Trapezoid" formation to a "Line" formation, and we control the quadrotor swarm in the "Line" formation through the obstacle area. Again, a few snapshots of the experiment video are shown in Figure 11.

VI. Conclusion Remarks

In this paper, we proposed a method for a human user to teleoperate a swarm of quadrotors from a standard joystick, while the quadrotors autonomously avoid collisions with obstacles and with each other. The intuitive human-swarm interface allows a single human user to operate an arbitrarily large quadrotor swarm. Using the framework of a Virtual Rigid Body, we successfully apply multiple vector fields among the quadrotor swarm, in both the global reference frame and the VRB local reference frame to avoid collisions by compliantly deforming the formation. Our algorithm assists the human user in flying quadrotor swarms through cluttered environments. The assistive collision avoidance method is validated in our experiments with a group of KMel

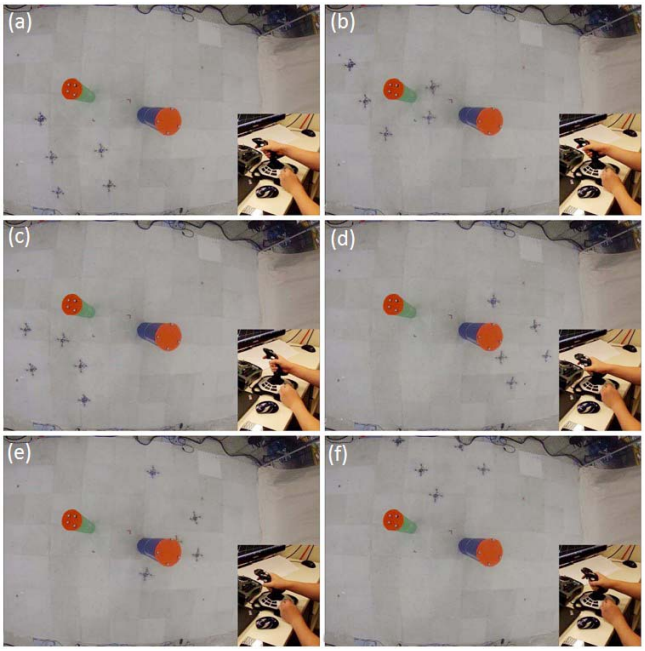


Fig. 10. A sequence of snapshots of experiment I. (a), The quadrotor swarm is initialized as a "Trapezoid" formation; (b), The quadrotor swarm significantly deforms while passing through obstacle 2 from two sides; (c), The corner of the "Trapezoid" formation is squeezed by obstacle 2; (d), The top quadrotor deviates from its desired position considerably; (e), One quadrotor comes close to obstacle 1, but with no collision; (f), The quadrotor swarm recovers its "Trapezoid" formation.

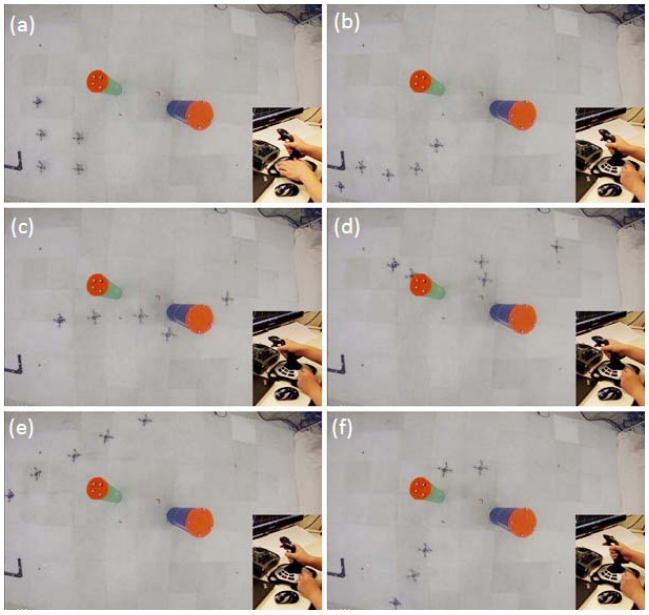


Fig. 11. A sequence of snapshots of experiment II. (a), The quadrotor swarm is initialized as a "Trapezoid" formation; (b), The VRB is in a transformation from the "Trapezoid" formation to a "Line" formation; (c), The "Line" formation is deformed by obstacle 1; (d), The quadrotor swam has successfully passed through the two obstacles; (e), The quadrotors are in a "Line" formation again; (f), the quadrotor swarm is significantly deformed by obstacle 2.

Nano Plus quadrotors in a motion capture system. In the future, we plan to adapt this method to work with on-board perception of the obstacles.

REFERENCES

[1] D. Zhou and M. Schwager, "Virtual rigid bodies for coordinated agile maneuvering of teams of micro aerial vehicles," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 1737–1742.

[2] ——, "Virtual rigid bodies for agile coordination of quadrotor swarms and human-swarm teleoperation," in *IEEE Transactions on Robotics (Under Review)*, 2015.

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

[4] D. Zhou and M. Schwager, "Vector field following for quadrotors using differential flatness," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 6567–6572.

[5] M. Turpin, N. Michael, and V. Kumar, "Capt: Concurrent assignment and planning of trajectories for multiple robots," *The International Journal of Robotics Research*, vol. 33, no. 1, pp. 98–112, 2014.

[6] ——, "Trajectory design and control for aggressive formation flight with quadrotors," *Autonomous Robots*, vol. 33, no. 1-2, pp. 143–156, 2012.

[7] E. Montijano, E. Cristofalo, D. Zhou, M. Schwager, and C. Sagues, "Vision-based distributed formation control without an external positioning system," *IEEE Transactions on Robotics (Under Review)*, 2015.

[8] A. Kushleyev, D. Mellinger, C. Powers, and V. Kumar, "Towards a swarm of agile micro quadrotors," *Autonomous Robots*, vol. 35, no. 4, pp. 287–300, 2013.

[9] J. Alonso-Mora, M. Schoch, A. Breitenmoser, R. Siegwart, and P. Beardsley, "Object and animation display with multiple aerial vehicles," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 1078–1083.

[10] S. Mastellone, D. M. Stipanović, and M. W. Spong, "Remote formation control and collision avoidance for multi-agent nonholonomic systems," in *Robotics and Automation (ICRA), 2007 IEEE International Conference on*. IEEE, 2007, pp. 1063–1067.

[11] S. Mastellone, D. M. Stipanović, C. R. Graunke, K. A. Intlekofer, and M. W. Spong, "Formation control and collision avoidance for multi-agent non-holonomic systems: Theory and experiments," *The International Journal of Robotics Research*, vol. 27, no. 1, pp. 107–126, 2008.

[12] A. Franchi, C. Masone, V. Grabe, M. Ryll, H. H. Bülthoff, and P. R. Giordano, "Modeling and control of uav bearing-formations with bilateral high-level steering," *The International Journal of Robotics Research*, vol. 31, no. 12, pp. 1504–1525, 2012.

[13] J. Alonso-Mora, S. Haegeli Lohaus, P. Leemann, R. Siegwart, and P. Beardsley, "Gesture based human-multi-robot swarm interaction and its application to an interactive display," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 5948–5953.

[14] A. M. Brandt and M. B. Colton, "Haptic collision avoidance for a remotely operated quadrotor uav in indoor environments," in *Systems Man and Cybernetics (SMC), 2010 IEEE International Conference on*. IEEE, 2010, pp. 2724–2731.

[15] X. Hou and R. Mahony, "Dynamic kinesthetic boundary for haptic teleoperation of aerial robotic vehicles," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. IEEE, 2013, pp. 4945–4950.

[16] J. Israelsen, M. Beall, D. Bareiss, D. Stuart, E. Keeney, and J. van den Berg, "Automatic collision avoidance for manually tele-operated unmanned aerial vehicles," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 6638–6643.

[17] J. Mendes and R. Ventura, "Assisted teleoperation of quadcopters using obstacle avoidance," *Journal of Automation Mobile Robotics and Intelligent Systems*, vol. 7, no. 1, pp. 54–58, 2013.

[18] A. Richards and J. P. How, "Aircraft trajectory planning with collision avoidance using mixed integer linear programming," in *American Control Conference, 2002. Proceedings of the 2002*, vol. 3. IEEE, pp. 1936–1941.

[19] D. Mellinger, A. Kushleyev, and V. Kumar, "Mixed-integer quadratic program trajectory generation for heterogeneous quadrotor teams," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 477–483.

[20] P. Martin, P. Rouchon, R. Murray *et al.*, "Flat systems, equivalence and trajectory generation. 3ème cycle," *Castro Urdiales (Espagne)*, p. 81, 2006.

[21] R. M. Murray, "Optimization-based control," *California Institute of Technology, CA*, 2009.