2018 IEEE International Conference on Robotics and Automation (ICRA)
May 21-25, 2018, Brisbane, Australia

Safe Teleoperation of Dynamic UAVs through Control Barrier Functions

Bin Xu[1] and Koushil Sreenath[2]

***Abstract*— This paper presents a method for assisting human operators to teleoperate highly dynamic systems such as quadrotors inside a constrained environment with safety guarantees. Our method enables human operators to focus on manually operating and flying quadrotor systems without the need to focus on avoiding potential obstacles. This is achieved with the presented supervisory controller overriding human input to enforce safety constraints when necessary. This method can be used as an assistive training solution for novice pilots to begin flying quadrotors without crashing them. Our supervisory controller uses an Exponential control barrier function based quadratic program to achieve safe human teleoperated flight. We demonstrate and validate our control approach through several experiments with multiple users with varying skill levels for three different scenarios of a quadrotor flying in a motion capture environment with virtual and physical constraints.**

I. INTRODUCTION

A. Motivatoin

Quadrotor UAVs (Unmanned Aerial Vehicles) are a well established research platform for dynamic mobility and have been widely used for innumerable research thrusts in the recent years. The advance of design, planning, estimation and control for these dynamic systems, makes quadrotors a versatile aerial robotic platform for many different tasks, including surveillance, bridge inspection, delivery and emergency rescue. Additionally, the sport of drone racing has also gained popularity because of the advances in quadrotor technology. However, the act of manually controlling a quadrotor with a remote controller is challenging for most people, especially for novice pilots. This is because the task of flying a quadrotor is too demanding for beginners, and in most cases the quadrotor ends up crashing during early flight experiments. Moreover, as pilots advance past the novice level, the tasks they attempt get more challenging - for instance the task of flying fast along a desired path with tight curves. To address the problem of enabling novice pilots to move up the skill scale, this paper focuses on a supervisory controller for safe teleoperation of a quadrotor. The resulting technique can also be applied to other dynamic systems that are typically hard for humans to teleoperate. The proposed methodology can also be used to provide low-level safety so as to enable applying learning algorithms onto dynamical systems.

[1]Bin Xu is with the Department of Mechanical Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, email: binx@andrew.cmu.edu.

[2]Koushil Sreenath is with the Department of Mechanical Engineering, University of California, Berkeley, CA 94720 email: koushils@berkeley.edu.

This work is supported in part by NSF grant CMMI-1538869 and in part by the Google Faculty Research Award.

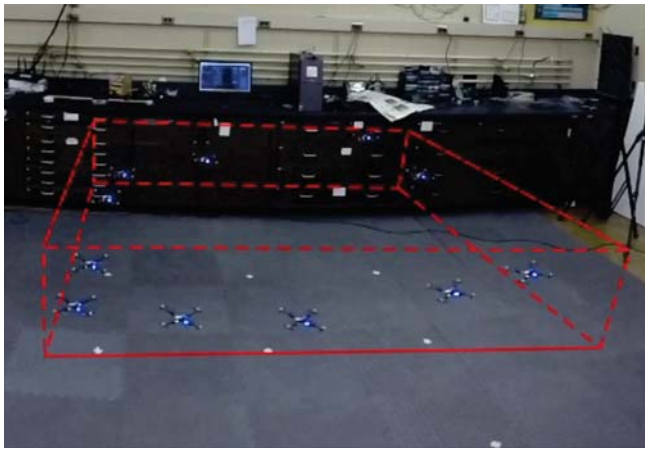

Fig. 1. A composite image showing our collision avoidance rectifier working on a manually teleoperated quadrotor. The supervisory safety rectifier follows the user input as closely as possible while overriding the user to ensure safety to ensure the quadrotor does not fly outside of the red virtual box. Such virtual safety constraints can help novice pilots learn to safely operate quadrotors and other dynamical systems. A video of the experiment is available at https://youtu.be/zVGPn6EK_qI.

B. Related Work

Safety in the form of collision avoidance of robotic systems has been widely explored in robotics research. Here we will list a few approaches used to achieve safe and collision-free motion in autonomous robots. Firstly, safety and collision avoidance can be enforced through trajectory planning. For instance, mixed-integer constraints are used in a Linear Program for safe trajectory generation in [1], where the quadrotor is approximated as a point mass. Mixed-integer constraints are also used in a quadratic program for safe trajectory generation for heterogeneous quadrotor teams in [2], wherein the optimization goal of the quadratic program is to minimize the square of the norm of the snap. Planning and collision avoidance have also been achieved using velocity constraints in [3] and acceleration constraints in [4]. However, trajectory generation methods typically are offline and online variants do not directly translate to high-dimensional systems.

Safety and collision avoidance can also be enforced through control. For instance, in [5], a vector-field following method is presented to avoid collisions by making the quadrotor follow a vector-field that is generated by taking into account the obstacles and the desired trajectory. More recently, quadratic programs formulated through control Lyapunov functions to guarantee stability and an augmented control barrier function to enforce safety in position space for a planar quadrotor is presented in [6]. This method has

also been extended to strictly guarantee time-varying safety-critical constraints for a 3D quadrotor [7]. Moreover, control barrier function based safety certificates have been proposed in [8] to achieve safe flight for quadrotors through trajectory re-planning, given the nominal trajectory and obstacles. However this early work did not consider manual human control of quadrotors. More recently, supervisory control using barrier functions has been used for safe autonomous control for automobiles in [9].

Apart from the use case of robot autonomy, safety requirements are also crucial for human teleoperation, such as in car driving [10], [11], manipulation of surgical robots [12], and quadrotor teleoperation. The main problem lies in a fluid interaction between the human operator's control input and the safety constraints. Teleoperation of robotic systems with safety constraints has been studied through haptic force feedback, wherein sensory information is provided to the operators in order to warn them so as to avoid violating safety constraints such as collisions [13]. However, the user could still choose to ignore the haptic feedback. Work in [14] addresses this by overriding the operator's input based on time-to-collision. Even though these methods can work under some simplistic situations, they completely ignore the dynamics of the system and can only be applied to specific safety constraints.

Another approach is to let human operators determine the trajectory during teleoperation. For instance, in [15], a human operator interacts with the system to select a trajectory that the low-level controller then tracks. Moreover, in [16], a quadrotor's future flight path is extrapolated based on the operator's inputs, which are then modified to avoid future collisions. Whereas, in [17], a swarm of quadrotors are collectively controlled by high-level operator input which the system uses to generate safe future collision-free motion using a vector-field following method. These methods consider the dynamics of the quadrotor, however the safe trajectory generation happens at discrete intervals, which could fail at higher speeds as generated paths become unsafe due to previously-unseen obstacles.

C. Contribution and Paper Structure

Our approach considers the quadrotor dynamics and the operator's control inputs directly in terms of desired roll, pitch, and thrust, to develop a safe control input rectifier. The main contributions of our work are:

- We formulate a direct method for safe human teleoperation of quadrotors based on exponential control barrier functions expressed through a quadratic program.
- We enable mediating between the human input and enforcing safety constraints through minimal modification of the human input.
- We present a detailed control design and experimental validation of our method for different human operators under three scenarios: teleoperated flight in (a) a virtual box, (b) a virtual tube, and (c) with real obstacles.

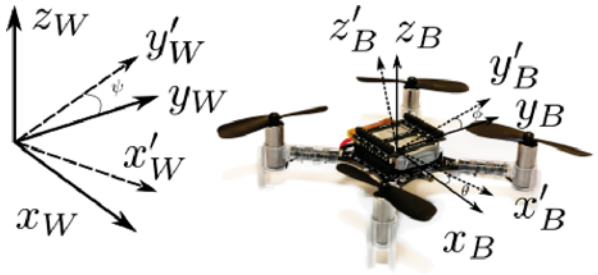

Fig. 2. World coordinate and body fixed coordinate of Crazyflie and Euler angles defined in these coordinates

D. Organization

The remainder of this paper is organized as follows. A math preliminary including quadrotor dynamics and exponential control barrier functions is introduced in Section II. We introduce control design to achieve safety with human operator teleoperation in Section III. The experimental setup, detailed controller design, results and quantification of operator performance will be discussed in Section IV. Finally, we present concluding remarks and discuss future work in Section V.

II. MATH PRELIMINARY

Having briefly introduced and motivated the objective of our work, we will now introduce some math preliminaries on quadrotor dynamics and exponential control barrier functions.

A. Quadrotor Dynamics

A quadrotor is a well-modeled underactuated dynamical system with forces and torques generated from four propellers and gravity. The relevant coordinate systems and free body diagram for the quadrotor are shown in Fig. 2. The world frame $\mathcal{F}_W$ is defined by axes x_W, y_W and z_W. The body fixed frame $\mathcal{F}_B$ is defined by axes x_B, y_B and z_B, in which x_B is the preferred forward direction and z_B is perpendicular to the plane of propellers pointing upward. Generally, quadrotors either have "x" or "+" configuration that relates their forward motion direction and the configuration of the propellers. In this paper, we use the Crazyflie micro quadrotor with the "x" configuration.

We use Z-X-Y Euler angles to define the roll (ϕ), pitch (θ) and yaw (ψ) angles between the body frame and the world frame. The Euler angles we use are defined in Fig. 2. The rotation matrix between the body fixed frame $\mathcal{F}_B$ and the world frame $\mathcal{F}_W$ is defined as,

$$ {}^{W}\boldsymbol{R}_B = \begin{bmatrix} c_\psi c_\theta - s_\phi s_\psi s_\theta & -c_\phi s_\psi & c_\psi s_\theta + c_\theta s_\phi s_\psi \\ c_\theta s_\psi + c_\psi s_\phi s_\theta & c_\phi c_\psi & s_\psi s_\theta - c_\psi c_\theta s_\phi \\ -c_\phi s_\theta & s_\phi & c_\phi c_\theta \end{bmatrix}, \tag{1} $$

where s_ϕ and c_ϕ denote $\sin(\phi)$ and $\cos(\phi)$, respectively and similarly for θ and ψ. The quadrotor dynamics given are then

as follows,

$$\begin{cases} \dot{\boldsymbol{r}} = \boldsymbol{v}, & (2) \\ \dot{\boldsymbol{v}} = \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} + \frac{1}{m}{}^{W}\boldsymbol{R}_B \begin{bmatrix} 0 \\ 0 \\ \sum_{i=1}^{4} F_i \end{bmatrix}, & (3) \\ \dot{\boldsymbol{R}} = \boldsymbol{R}\boldsymbol{\Omega}, & (4) \\ \boldsymbol{J}\dot{\boldsymbol{\Omega}} = \boldsymbol{M} - \boldsymbol{\Omega} \times \boldsymbol{J}\boldsymbol{\Omega}, & (5) \end{cases}$$

where $\boldsymbol{r}$ denotes the position vector of the center-of-mass of the quadrotor in the world frame $\mathcal{F}_W$, F_i denotes the scalar force generated by the i-th rotor, $\boldsymbol{\Omega}$ and $\boldsymbol{M}$ are the angular velocity and torque of the quadrotor in the body frame, and $m, \boldsymbol{J}$ are respectively the mass and the inertia matrix of the quadrotor.

For simplicity, we use the small angle assumption, which assumes that the Euler angles would be less than 10 degrees. In this case, we can get $\sin(\phi) \approx \phi$, and $\cos(\phi) \approx 1$ (similar relations hold for θ and ψ.) With this assumption, the nonlinear dynamics in (3) are transformed as, see [18],

$$\begin{aligned} \ddot{r}_1^{des} &= g(\theta^{des}\cos\psi + \phi^{des}\sin\psi), \\ \ddot{r}_2^{des} &= g(\theta^{des}\sin\psi - \phi^{des}\cos\psi), \\ \ddot{r}_3^{des} &= \frac{F^{des}}{m} - g. \end{aligned} \tag{6}$$

Here F^{des} is desired thrust that is to be generated as the sum of all the individual propeller forces.

The roll and pitch angles for the attitude controller as well as the total thrust can be represented in terms of the desired acceleration by inverting the above relationship, to obtain,

$$\begin{aligned} \phi^{des} &= \frac{1}{g}(\ddot{r}_1^{des}\sin\psi - \ddot{r}_2^{des}\cos\psi), \\ \theta^{des} &= \frac{1}{g}(\ddot{r}_1^{des}\cos\psi + \ddot{r}_2^{des}\sin\psi), \\ F^{des} &= m(\ddot{r}_3^{des} + g). \end{aligned} \tag{7}$$

In this paper, the human operator's input comprises of the desired Euler angles θ^{des}, ϕ^{des}, and the the desired thrust F^{des} (the desired yaw angle is assumed to be zero). These quantities are than transformed into desired translational accelerations through (6), which then makes the acceleration of the quadrotor as direct control input $\boldsymbol{u}$ that is specified by the user, where

$$\boldsymbol{u} = \begin{bmatrix} \ddot{r}_1^{des} \\ \ddot{r}_2^{des} \\ \ddot{r}_3^{des} \end{bmatrix}, \tag{8}$$

leading to a second-order integrator system,

$$\begin{bmatrix} \dot{\boldsymbol{r}} \\ \ddot{\boldsymbol{r}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{I} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{r} \\ \dot{\boldsymbol{r}} \end{bmatrix} + \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{1} \end{bmatrix} \boldsymbol{u}. \tag{9}$$

Remark 1: Since the human operator's input are the desired Euler angles, that are then transmitted to the onboard controller, our proposed safety rectifier minimally modifies these inputs so as to guarantee the enforcement of the safety constraints. Consequently, the dynamics seen by our proposed controller is that in (9). As a result, we see the quadrotor as a point mass.

B. Exponential Control Barrier Function

We address the collision avoidance problem using Exponential Control Barrier Functions (ECBFs) [20]. A control Barrier function (CBF) is a Control Lyapunov Function (CLF)-like function and is defined over the state-space. While a CLF is used to guarantee stability, a CBF is used to guarantee a safety constraint. An ECBF is an extension of the CBF that provides two key advantages - the ability to address higher relative-degree safety constraints as well as the ability to enforce these constraints through linear control theory techniques such as pole placements. In this paper, we leverage both these advantages of the ECBF.

Suppose we have a continuously differentiable function $h(t, x)$, the super level set can be defined as $\mathcal{C}_t = \{x \in \mathbb{R}^n \mid h(t, x) \geq 0\}$. This is the safe set representing the subset of the state-space that is considered safe. The goal of our controller is to strike a balance between teleoperation and keeping the system state within this set, i.e., to guarantee the forward invariance of this set. The ECBF offers a way to reliably guaratnee forward invariance of the safe set. As we will see next, through enforcement of the ECBF constraint, $h(t, x)$ would always remain nonnegative, thereby achieving safety, i.e., $x(t) \in \mathcal{C}_t, \forall t \geq 0$.

For the dynamical system in (9), comprising of a double integrator chain, a function $h(t, x)$ that is only dependent on the Cartesian position $\boldsymbol{r}$ has relative degree 2 [21], i.e., only the second time-derivative of h depends on the control input. Then, $h(t, x)$ is a ECBF for the dynamical system (9) if there exists a $\boldsymbol{K} \in \mathbb{R}^2$ that places the poles of $\ddot{h} + \boldsymbol{K} \cdot [h \quad \dot{h}] = 0$ on the negative real line, and a control input u that satisfies the inequality,

$$\ddot{h}(t, x, u) + \boldsymbol{K} \cdot [h(t, x) \quad \dot{h}(t, x)]^T \geq 0, \forall x \in \mathcal{C}_t. \tag{10}$$

This ensures that the control input drives the system state $x(t)$ to ensure forward invariance of $\mathcal{C}_t$ and thereby guarantees the enforcement of the safety constraint $h(t, x) \geq 0$.

III. CONTROL METHOD DESIGN

Having introduced some math preliminaries above, we will now delve into our proposed safety rectifier control design that minimally modifies the user input to enforce safety.

Let us consider the human operator controlling the quadrotor. The normal control method is to have the operator provide four inputs, the desired roll, pitch, yaw rate and thrust, through pushing the joystick on a remote controller. For the human operator to fly a quadrotor with safety, he/she needs to perceive the surrounding environment and according to the information gathered, decide how to control the quadrotor to stay in the safe region by modifying the above four inputs. This task is always demanding for a beginner, and in most cases, the quadrotor crashes into the environment.

In order to achieve the goal of safe teleoperation, we convert he human operator's inputs to the desired direction

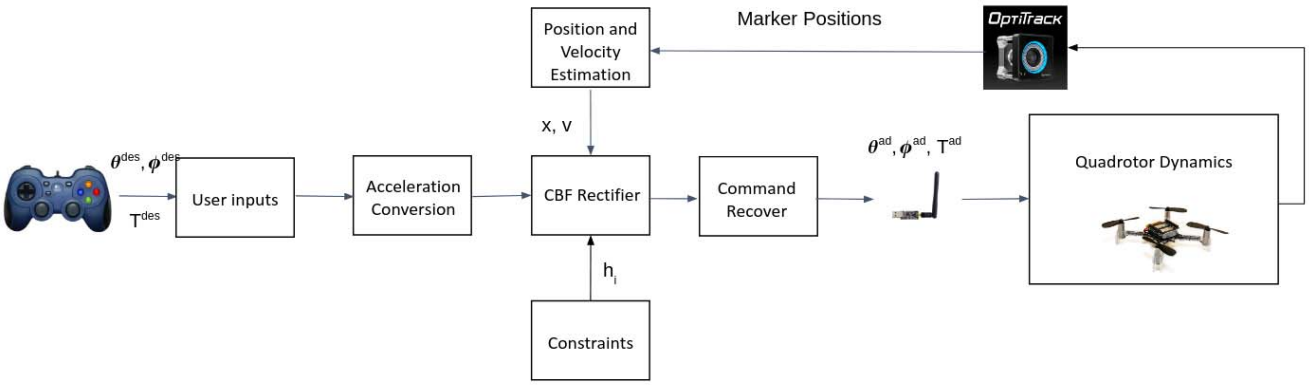

Fig. 3. A control block diagram illustrating the user joystick input, the proposed controller, and a quadrotor in a motion capture environment. The human operator provides the desired roll, pitch, and thrust through a joystick (the yaw is held constant at zero for simplicity.) The user inputs are converted to desired Cartesian accelerations of the quadrotor through (6) - the *Acceleration Conversion* block. Our supervisory controller minimally rectifies these accelerations to meet the safety constraints specified by each of the h_i by minimizing the difference between the user accelerations and the modified accelerations through the supervisory controller (11) - the *CBF Rectifier* block. The rectified accelerations are then converted to desired roll, pitch, and thrust through (7) - the *Command Recover* block - and then sent to the quadrotor's onboard attitude controller.

and acceleration of the quadrotor through (6). Then the problem would be converted to a simpler problem, that is with the given direction and acceleration, determine whether to modify the inputs specified by human operator to enforce the safety constraints in a supervisory manner. Afterwards the new control commands can be generated based on the updated admissible direction and acceleration, and sent to the onboard quadrotor attitude controller. Thus, the controller serves as a supervisory controller to provide a safety check in the whole control process. Note that the proposed method minimally modifies the human operator inputs based on an online optimization to guarantee safety. The proposed controller can also be used as a low-level supervisory controller that provides safety while applying various learning algorithms onto dynamical systems.

More specifically, after computing the desired acceleration through (6) based on the human operator's desired roll, pitch, thrust inputs, the Collision Avoidance Rectifier executes a constrained quadratic optimization to minimally modify the accelerations to satisfy the safety constraints. To design the controller for collision avoidance, we need first define a safety set as $\mathcal{C}_t = \{x \in \mathbb{R}^n | h(x,t) \geq 0\}$ for quadrotor teleoperation, in which $h(x,t)$ is the ECBF for this problem. ECBF is used as a constraint in the Quadratic Program (QP), which is used to minimize the difference between the admissible control inputs and original control inputs. The general form is defined as

$$\begin{aligned} \hat{\boldsymbol{u}}^* = &\underset{\hat{u}_i}{\operatorname{argmin}} ||\hat{u} - u||^2 \\ s.t. \quad &\ddot{h} + \boldsymbol{K} \cdot [h \quad \dot{h}]^T \geq 0, \end{aligned} \tag{11}$$

where h is as a continuously differentiable scalar function that describes the safety constraint to be enforced. The QP ensures that the desired acceleration would not be modified if the quadrotor satisfies the safety constraints, and would be minimally modified if violating the safety constraints. The control inputs $\hat{\boldsymbol{u}}$ generated from QP can then be converted to roll, pitch and thrust according to (7) and transmitted to the quadrotor. See Fig. 3 for a block diagram representation of our system.

Remark 2: The control method presented in this paper is similar to [8] where a ECBF is used to guarantee safety flight for trajectory tracking problem of quadrotors. However, in our work, we consider a human operator and study the problem of teleoperation. Moreover, since we consider the human operator input, our problem is now of relative-degree 2, simplifying the CBF rectifier. Our work also presents the formulation in a supervisory role by minimizing the deviation from the human input.

IV. EXPERIMENTAL RESULTS

Having presented our proposed control method design, we now test our framework under 3 different scenarios, including (1) teleoperated flight inside a box, (2) teleoperated flight inside a tube and (3) teleoperated flight with physical obstacles. These scenarios will help illustrate the performance of our method. We draw inspiration from drone racing, with Scenario 1 designed for teaching basic flight, Scenario 2 for teaching teleoperation for flight along a trajectory (we can build tubes along any desired trajectory) and Scenario 3 to teach flying without hitting obstacles. In this section, we will provide more details on the experimental setup, a procedure to design specific controller for each scenario, results from experiments, and one approach to quantify user's skill and performance based on logged experimental data.

A. Experiment Setup

We use a Crazyflie 2.0 miniature quadrotor UAV as the experimental platform that is teleoperated using a Logitech Gamepad F310 by a human operator. The crazyflie_ros package released by ACTLab at USC [22] provides a simple PID

position controller compatible with the OptiTrack Motion Capture System and also provides an interface to manually control the Crazyflie with a gamepad as a remote controller. See Fig. 3 for the experimental setup. The provided interface reads in the user control inputs and then sends it directly to the Crazyflie as control commands through a Crazyradio dongle. Our method is implemented based on this interface to achieve safe human teleoperation.

We will use the CrazyRadio Dongle to send the output commands for the quadrotor. We next present some low-level details of the Crazyflie. The thrust generated by any of the rotors can be expressed in this form [19]

$$F_i = K_F \omega_i^2, \tag{12}$$

where K_F is a non-dimensional thrust parameter and ω_i is the angular velocity of the i-th rotor. According to Crazyflie's firmware, the voltage sent to each DC motor is controlled using a pulse width modulation (PWM) signal specified as a 16-bit number, ranging from 0 to 65535, which means that the direct input to this system is a PWM signal [23]. So angular velocity ω_i can be expressed with PWM. Combining this with (12), we can generate a mapping from PWM to thrust. This relationship can be approximated by a quadratic function through system identification tested on one specific Crazyflie. The mapping is provided in [24] as

$$\begin{aligned} F_i = {} & 2.130295 \times 10^{-11} \cdot PWM^2 \\ & + 1.032633 \times 10^{-6} \cdot PWM + 5.484560 \times 10^{-4}. \end{aligned} \tag{13}$$

Apart from the Crazyflie 2.0, the Crazyradio dongle, and the Logitech Gamepad, we also use a OptiTrack motion capture system in our experiments. This provides the relative position of the quadrotor with respect to the obstacles. The relative velocity of the quadrotor is estimated from the position information provided from the motion capture system [25]. Note that since the human operator is actually flying the quadrotor, we do not need absolute position information of the quadrotor, but only relative pose information with respect to obstacles. This makes our methodology potentially transferable for outdoor flight where a camera system detects the relative location of the obstacles with respect to the quadrotor.

The control block diagram is shown in Fig. 3. In our setup, the desired yaw angle is set to be zero for simplicity. Thus the human operator has three control inputs for teleoperation, which are the desrired roll and pitch angles as well as the thrust, with the yaw rate controlled by a PID controller to keep the yaw zero. All four control inputs will be sent to the onboard attitude controller through the Crazyradio dongle. As is shown in the control block diagram, the output from the CBF rectifier will be converted to control commands required by the attitude controller through the "Command Recover block".

B. Scenario Specific Controller Design

In this part we present the controller design for each scenario. Note that the specific control methods for these three scenarios only differ in the barrier function that is used, i.e., the safety region. The basic idea is still the same as discussed earlier. We will present the procedure for Scenario 1 in detail and mention how to define the barrier functions for Scenarios 2 and 3.

For the Scenario 1, teleoperated quadrotor flight inside a virtual box, the hope is to assist a beginner pilot achieve a basic safe flight in a 3D space. Note that for this case, there will be six constraints as specified in (14), corresponding to the six faces of the virtual box, and all of these constraints are decoupled. These are,

$$\begin{aligned} r_{1min} \le r_1 \le r_{1max} \\ r_{2min} \le r_2 \le r_{2max} \\ r_{3min} \le r_3 \le r_{3max}. \end{aligned} \tag{14}$$

Users can specify the minimum and maximum values for these constraints according to the actual environment for the flight test. To satisfy the constraints, the safety set $\mathcal{C}$ is defined as

$$\begin{aligned} \mathcal{C} &= \{r_i \mid h_j^i(r_i) \ge 0\}, \quad i = 1,2,3, \quad j = 1,2, \\ h_1^i(r_i) &= r_{imax} - r_i, \\ h_2^i(r_i) &= r_i - r_{imin}, \end{aligned} \tag{15}$$

where r_1, r_2, r_3 stand for x, y and z direction respectively. To ensure the forward invariance of the safe set $\mathcal{C}$, the control inputs should satisfy (10). The safety barrier constraints can be rearranged into linear constraints on the acceleration,

$$\boldsymbol{A} \cdot \boldsymbol{u} \le \boldsymbol{b}, \tag{16}$$

where

$$\boldsymbol{A} = \begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & -1 \end{bmatrix}, \quad \boldsymbol{b} = \boldsymbol{K} \cdot [h, \dot{h}].$$

The safety constraints can be then incorporated into a quadratic program (11) to minimize the deviation between the desired accelerations and the modified accelerations which satisfy the safety constraints. This method would respect human operator's control input as close as possible while forcing the quadrotor to stay inside the safe region. The desired roll, pitch and thrust can then be computed from (7). Note that we only have modified total thrust $\hat{F}$ got from the CBF rectifier, not the control input PWM for the quadrotor. The characteristic of DC motor of the quadrotor would be used to calculate the PWM inversely. The inverse calculation here would be to simply solve a quadratic equation according to (13). Thus the actual control inputs, roll, pitch and PWM can be recovered from the modified acceleration. With this method presented, the quadrotor would satisfy flying inside a virtual box without violating the boundary, and this will be demonstrated in the experiments subsection.

Note that for the two other scenarios, the idea of control method is the same while the barrier functions to capture the constraints are different. We only provide the constraints and

safety sets used in the experiments. This just leads to only differences in the constraints of the QP in (11). For Scenario 2, we provide the constraints in (17) to approximate the tube.

$$\begin{aligned} R_{min}^2 \leq r_1^2 + r_2^2 \leq R_{max}^2, \\ r_{3min} \leq r_3 \leq r_{3max}. \end{aligned} \tag{17}$$

These constraints serve as a virtual tube that the quadrotor would fly inside, where the human operator has freedom to maneuver. This setup aims to help beginners to control a quadrotor to fly a circle, which is often difficult for them to do. The safety sets in this case would be

$$\begin{aligned} \mathcal{C} &= \{r_i \mid h_j(\boldsymbol{r}) \geq 0\}, \quad i = 1,2,3, \quad j = 1,2,3,4, \\ h_1(\boldsymbol{r}) &= R_{max}^2 - r_1^2 - r_2^2, \\ h_2(\boldsymbol{r}) &= r_1^2 + r_2^2 - R_{min}^2, \\ h_3(\boldsymbol{r}) &= r_{3max} - r_3, \\ h_4(\boldsymbol{r}) &= r_3 - r_{3min}. \end{aligned} \tag{18}$$

For Scenario 3, we use a super-ellipsoid to represent the obstacle, and make the quadrotor not collide with that obstacle by applying the constraint through a quadratic program. Through this example, we want to demonstrate that our method can also work for real obstacles and not only for virtual obstacles. The general implicit equation for a super-ellipsoid is, see [26],

$$\left[(\frac{x}{a})^r + (\frac{y}{b})^r \right]^{\frac{n}{r}} + (\frac{z}{c})^n \leq 1. \tag{19}$$

To better characterize the size of the obstacle, we would use $r = n = 4$ [8]. The safety set would then be

$$\begin{aligned} \mathcal{C} &= \{r_i \mid h(\boldsymbol{r}) \geq 0\}, \quad i = 1,2,3, \\ h(\boldsymbol{r}) &= (\frac{r_1}{a})^4 + (\frac{r_2}{b})^4 + (\frac{r_3}{c})^4 - d_s, \end{aligned} \tag{20}$$

where a, b, c are chosen based on the shape of obstacles and d_s is the safety distance to obstacles based on size of the quadrotor and the tracking error. With the safety sets defined above, the controller design procedure would be identical with the above. In this way the control method can guarantee safe flight.

C. Experiments

1) Teleoperated Flight Inside a Virtual Box: This scenario requires the safety rectifier to modify the control inputs from the human operator when the quadrotor could violate the safety requirements, i.e., have sufficient velocity to exit the box, else leave the human inputs unchanged. Human operator can operate in a box with range from -1 m to 1 m in x and y direction, 0.6 m to 1 m in z direction. Fig. 4 shows the original control inputs and modified control inputs got from the safety rectifier. As is shown in the figure, user defined control inputs are left unchanged if the inputs are admissible and be overridden if violating the safety boundary. The control inputs are be overridden if the system is likely to violate the safety constraints. Fig. 5 shows the trajectories of the quadrotor teleoperated by three different human operators arbitrarily in 3D space. The teleoperation starts from the center of the virtual box, as is shown in the figure, and the trajectory never exceeds the virtual bounds (blue box) we defined.

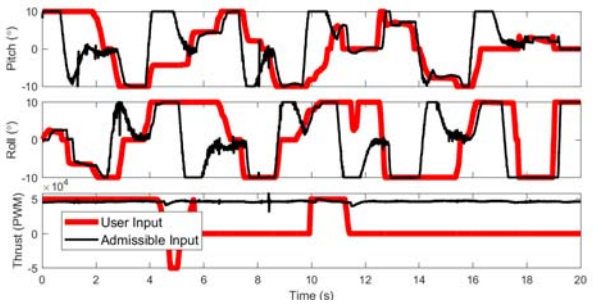

Fig. 4. User control inputs and admissible control inputs generated from CBF Rectifier. The output of the CBF rectifier is identical to the human input when the safety constraint is not likely to be violated. The human input is minimally overridden when the human input could cause the system to violate the constraint.

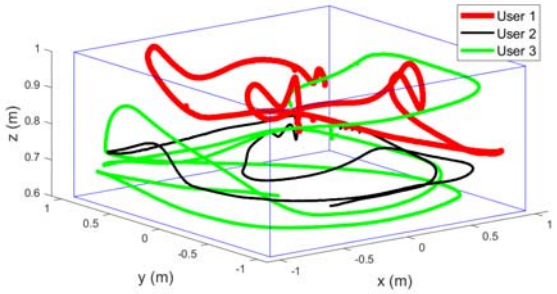

Fig. 5. 3D Space Trajectories of three different human operators teleoperating the quadrotor to fly in a virtual box with the help of our method. As can be seen, all the trajectories are within the virtual box. Snapshots of one quadrotor flight are shown in Fig. 1.

2) Teleoperated Flight Inside a Virtual Tube: This scenario forces the quadrotor to fly inside a circular track along with bounds in z direction. The circular track has inner radius of 0.6 m and outer radius 0.8 m. Just like the first scenario, when the quadrotor has a velocity that could cause it to exceed the bounds, the rectifier would override the control inputs and force the quadrotor to stay inside the track. This experiment helps users to get familiar with flying a quadrotor in a circle, which is even harder than Scenario 1. To get a more general conclusion, we ask several human operators to do tests to demonstrate the performance of this method. Fig. 6 shows the trajectories of quadrotor teleoperated by five different human operators in x-y plane. As expected, the five trajectories differ from each other, however they all stay inside the circular track. Note that the z direction has the same bounds as Scenario 1 so we do not include the plot for the z direction here.

3) Teleoperated Flight with a Physical Obstacle: This scenario incorporates the safety rectifier to help human operators avoid real obstacles in order to test the performance of our method. Fig. 7 shows how the quadrotor reacts when directly pushing it to the obstacle. Here we only plot out

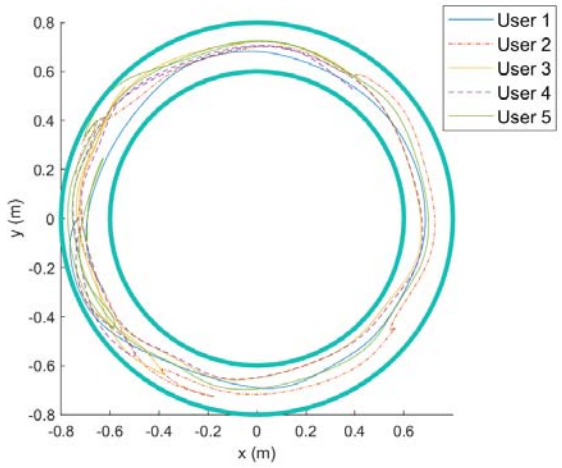

Fig. 6. Planar trajectories of five different human operators teleoperating the quadrotor to fly within a virtual cylinder with the help of our control method.

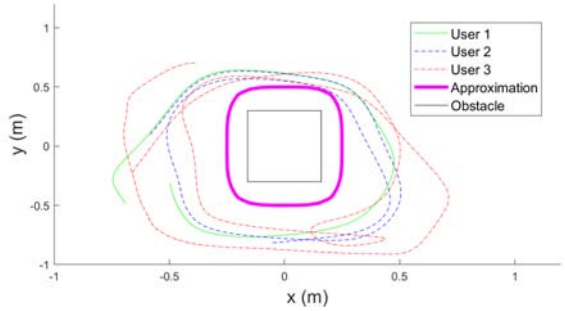

Fig. 7. Planar trajectories of three different human operators teleoperating the quadrotor to fly while avoiding a physical obstacle with the help of our method. The subjects tried to get the quadrotor to collide with the obstacle, but our system prevented the collision. This can be seen when the quadrotors were piloted close to the obstacle in the upper right and also when the operators attempted to fly directly into the obstacle at the bottom.

the trajectory in xy plane. The obstacle is a box and we use a super-ellipsoid to approximate its shape, with $a = 1$, $b = 2$, $c = 4$ and safety distance $d_s = 0.25$ m in (20). The safety distance is chosen based on the shape of the obstacle, size of quadrotor, tracking error, and the configuration of quadrotor. In the calculation, we only use the projection of the super-ellipsoid onto the x-y plane, which is shown in Fig. 7. We plot 3 different user's teleoperated trajectories in the figure and all these trajectories stay inside the safe region we defined, i.e., outside the obstacle. Note that when doing the experiment, we deliberately add bounds which is same as Scenario 1 in case a novice user causes the quadrotor to fly too far and ends up crashing into a wall.

D. Quantification of User Skill and Performance

Based on the experiments, we introduce a quantity to quantify the human operator's flight skill experience according to the differences in the original control inputs and admissible control inputs (computed by the CBF rectifier).

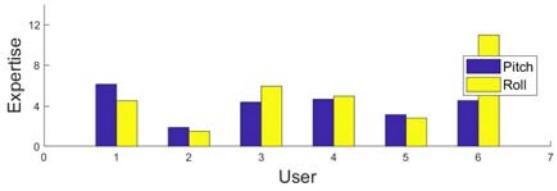

Fig. 8. Quantification of flight skill expertise for human operators teleoperating the quadrotor in a virtual box. Small values indicate better expertise since in this case the human operators fly safely so as to not get the CBF rectifier to override their input.

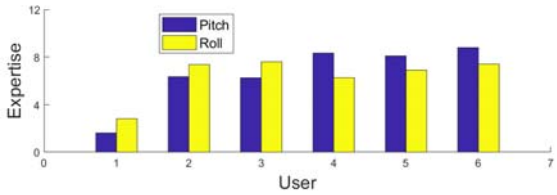

Fig. 9. Quantification of flight skill expertise for human operators teleoperating a quadrotor in a tube. Lower values indicate better expertise.

This can be calculated as

$$expertise = \frac{\int_{t_1}^{t_2} ||\hat{u} - u||^2 dt}{t_2 - t_1}, \tag{21}$$

where u is the three human control inputs and $\hat{u}$ is the control inputs computed by the CBF rectifier in (11). This quantity will be small if the operator is an expert and can enforce the safety constraints for the task on his own without having the supervisory controller override his control. This quantity will be relatively large if the operator is a novice pilot of the quadrotor since in this case the supervisory controller overrides his input often. Fig. 8 shows the expertise in roll and pitch for 6 human operators teleoperating the quadrotor inside the virtual box (Scenario 1). Fig. 9 shows the expertise in roll and pitch for 6 human operators teleoperating the quadrotor along a circle (Scenario 2). From the two plots, it is clearly shown that the human operator skill levels for flying in a circle are lower than when flying within a box. Note that the two experiments are done separately, and the user id's do not align between experiments. We hypothesize that the expertise metric can further help people know how well they are improving between multiple practice rounds of teleoperating such systems.

V. CONCLUSIONS AND FUTURE WORK

This paper presents a method for assisting human operators to teleoperate dynamic quadrotors inside constrained environments with safety guarantees. The presented supervisory controller uses exponential control barrier functions (ECBFs) formulated in a quadratic program to enforce safety guaratnees while minizing the difference between the human input and the controller commands to the quadrotor. Three experimental scenarios are presented to validate the controller. Our controller can potentially be used as a safe training tool to improve the skills of a pilot operator at any skill level. Moreover, more complex task scenarios

with complex safety constraints can also be setup beyond the simple tasks presented here. Furthermore, the proposed methodology can also be used to provide low-level safety for higher-level algorithms such as reinforcement learning.

A few drawbacks of the current work involve the requirement of a motion capture environment, and the linearization of quadrotor dynamics for the command rectifier. The method can potentially be extended to flight outside the motion capture system environment with the presence of onboard sensors that provide relative distance and velocity estimates to obstacles. Moreover, recent work on augmented control barrier functions in [7] completely removes the requirement of integrating a linearized dynamical model for the command rectifier. These extensions to address the shortcomings of this work will be part of future work.

ACKNOWLEDGMENT

B. Xu would like to thank G. Wu for helping understand much of the math behind safety-critical control as well as in running experiments and post-processing data.

REFERENCES

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

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

[3] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *The International Journal of Robotics Research*, vol. 17, no. 7, pp. 760–772, 1998.

[4] J. Van Den Berg, J. Snape, S. J. Guy, and D. Manocha, "Reciprocal collision avoidance with acceleration-velocity obstacles," in *IEEE International Conference on Robotics and Automation*, 2011, pp. 3475–3482.

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

[6] G. Wu and K. Sreenath, "Safety-critical control of a planar quadrotor," in *American Control Conference*, 2016, pp. 2252–2258.

[7] ——, "Safety-critical control of a 3d quadrotor with range-limited sensing," in *ASME Dynamic Systems and Control Conference*, 2016, pp. V001T05A006–V001T05A006.

[8] L. Wang, A. D. Ames, and M. Egerstedt, "Safe certificate-based maneuvers for teams of quadrotors using differential flatness," *arXiv preprint arXiv:1702.01075*, 2017.

[9] Y. Chen, H. Peng, and J. Grizzle, "Obstacle avoidance for low-speed autonomous vehicles with barrier function," *IEEE Transactions on Control Systems Technology*, vol. 26, no. 1, pp. 194–206, 2018.

[10] J. C. McCall and M. M. Trivedi, "Human behavior based predictive brake assistance," in *IEEE Intelligent Vehicles Symposium*, 2006, pp. 8–12.

[11] J. Cao, H. Liu, P. Li, and D. J. Brown, "State of the art in vehicle active suspension adaptive control systems based on intelligent methodologies," *IEEE transactions on intelligent transportation systems*, vol. 9, no. 3, pp. 392–405, 2008.

[12] M. Dewan, P. Marayong, A. M. Okamura, and G. D. Hager, "Vision-based assistance for ophthalmic micro-surgery," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2004, pp. 49–57.

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

[14] 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.

[15] C. Masone, A. Franchi, H. H. Bülthoff, and P. R. Giordano, "Interactive planning of persistent trajectories for human-assisted navigation of mobile robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 2641–2648.

[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 *IEEE International Conference on Robotics and Automation*, 2014, pp. 6638–6643.

[17] D. Zhou and M. Schwager, "Assistive collision avoidance for quadrotor swarm teleoperation," in *IEEE International Conference on Robotics and Automation*, 2016, pp. 1249–1254.

[18] D. Mellinger, "Trajectory generation and control for quadrotors," Ph.D. dissertation, University of Pennsylvania, 2012.

[19] E. Greitzer, Z. Spakovszky, and I. Waitz, "Thermodynamics and propulsion," *Mechanical Engineering, MIT*, 2006.

[20] Q. Nguyen and K. Sreenath, "Exponential control barrier functions for enforcing high relative-degree safety-critical constraints," in *American Control Conference*, 2016, pp. 322–328.

[21] H. K. Khalil, *Nonlinear control*. Prentice Hall, 2014.

[22] W. Hoenig, C. Milanes, L. Scaria, T. Phan, M. Bolas, and N. Ayanian, "Mixed reality for robotics," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015, pp. 5382–5387.

[23] C. Luis and J. L. Ny, "Design of a trajectory tracking controller for a nanoquadcopter," *arXiv preprint arXiv:1608.05786*, 2016.

[24] J. Förster, "System identification of the crazyflie 2.0 nano quadrocopter," Masterss Thesis, ETH Zurich, 2015.

[25] S. Diop, J. Grizzle, P. Moraal, and A. Stefanopoulou, "Interpolation and numerical differentiation for observer design," in *American Control Conference*, vol. 2, 1994, pp. 1329–1329.

[26] A. H. Barr, "Superquadrics and angle-preserving transformations," *IEEE Computer graphics and Applications*, vol. 1, no. 1, pp. 11–23, 1981.