2016 IEEE International Conference on Robotics and Automation (ICRA)
Stockholm, Sweden, May 16-21, 2016

Safe and Robust Robot Maneuvers Based on Reach Control

Marijan Vukosavljev, Ivo Jansen, Mireille E. Broucke, and Angela P. Schoellig

Abstract— **In this paper, we investigate the synthesis of piecewise affine feedback controllers to address the problem of safe and robust controller design in robotics based on high-level controls specifications. The methodology is based on formulating the problem as a collection of reach control problems on a polytopic state space. Reach control has so far only been developed in theory and has not been tested experimentally on a real system before. Using a quadrocopter as our experimental platform, we show that these theoretical tools can achieve fast, albeit safe and robust maneuvers. In contrast to most traditional control techniques, the reach control approach does not require a predefined open-loop reference trajectory or spacial path. Experimental results on a quadrocopter show the effectiveness and robustness of this control approach. In a proof-of-concept demonstration, the reach controller is implemented in one translational direction while the other degrees of freedom are stabilized by separate controllers.**

I. Introduction

This paper proposes a novel framework for control of complex robotic maneuvers that simultaneously impose requirements of safety, fast response, and a desired sequence of events. We apply the framework to a simple side-to-side maneuver on a quadrocopter in order to expose the main features of the framework. The framework is based on using hybrid systems, event-based switching, and solving a collection of so-called reach control problems (RCP). The RCP has an extensive theoretical development, see [10]–[13], [15], but it has been completely lacking in experimental validation. The primary goal of this paper is to illustrate, for the first time on a real system, the various strengths offered by the reach control approach.

Because our main application is quadrocopter maneuvering, we give an overview of current approaches, particularly placing our reach control approach within this literature. There are two predominant methods for control of quadrocopter maneuvers: timed trajectory tracking and path following. In timed trajectory tracking, an open-loop reference trajectory as a function of time is predefined. Then a controller that stabilizes the system to the trajectory is designed. For example, in [3], an impressive collection of aggressive quadrocopter maneuvers is featured using this approach. Timed trajectory tracking is the most common method in the literature [1]–[3]. On the other hand, difficulties arise in finding the open-loop reference trajectory. For example, in [1] it is first geometrically specified using splines, and then time parameterized so that the resulting reference trajectory is feasible. Also, while timed trajectory tracking can provide high-performance maneuvers, due to the open-loop nature of the reference signals, any additional unaccounted disturbances can quickly deteriorate performance.

In path following, a 3D spatial (untimed) path is specified in output space. Then an output stabilization method is used to keep the system on the path. Recent applications of path following to quadrocopter maneuvering include [5]–[7]. The benefit of this method compared to timed trajectory tracking is better control of transients: when the system deviates from the path, it must only steer back to the path rather than to a specific point in time. The difficulty in this method is again finding paths feasible for the dynamics and constraints.

In our approach, it is not required to generate a feasible timed trajectory or path. Rather, control specifications are given that restrict the dynamics to a feasible region of the state space. Further, these specifications inform on how the state must evolve in that region. The region is then partitioned into smaller regions (in our case simplices) and a feedback controller is designed for each region to guarantee correct evolution of the state. As such, our method has several significant advantages: 1) we bypass the construction of a reference trajectory or path; 2) we obtain feedback controllers, not open-loop controls; 3) we obtain controllers for the entire feasible region of operation, not only a neighborhood of a path; 4) we explicitly account for safety constraints and actuator limits. The main difficulty of our method (as it is implemented right now) is its application to high-dimensional systems, where the required state-space partitioning becomes more involved. In the future, advanced computational tools may be adopted from other fields to address this problem. On the other hand, this paper shows that high-order models can be reduced to simpler models with no degradation of performance. For related methods to our approach see [8], [9]. Reach control has never been applied on a real system before. This work presents a proof-of-concept of its practical applicability.

II. Methodology

In this section, we briefly outline the reach control methodology, which allows us to define high-level controls specifications, and results in safe and robust system behavior.

Marijan Vukosavljev and Mireille E. Broucke are with the Dept. of Electrical and Computer Engineering, University of Toronto, Canada (e-mails: mario.vukosavljev@mail.utoronto.ca, broucke@control.utoronto.ca). Ivo Jansen was a visiting student from the Dept. of Mechanical Engineering, Eindhoven University of Technology, the Netherlands (email: i.h.m.jansen@student.tue.nl). Angela P. Schoellig is with the University of Toronto Institute for Aerospace Studies (UTIAS), Canada (email: schoellig@utias.utoronto.ca). Supported by the Natural Sciences and Engineering Research Council of Canada (NSERC).

Associated video at: http://tiny.cc/quadrotorRCPx.

978-1-4673-8026-3/16/$31.00 ©2016 Crown

A. System Model and Control Specifications

We assume that the system is modeled as a finite-dimensional, dynamical system, with possibly nonlinear dynamics:

$$\dot{\tilde{s}} = \tilde{f}(\tilde{s}, \tilde{u}), \tag{1}$$

where $\tilde{s} \in \mathbb{R}^n$ is the state and $\tilde{u} \in \mathbb{R}^{n_u}$ is the control input.

The objective of the controller is defined in terms of high-level safety and event specifications. In particular, the approach can handle the following types of specifications:

a) Safety and Liveness: Safety and liveness constraints define the region in the state space that the system is allowed to visit. Safety constraints limit the dynamics to a safe regime of operation. Liveness constraints enforce fast, lively response. Our framework requires that the feasible region be a polytope $\mathcal{P}$, as shown in Figure 4.

b) Desired Temporal Sequence: This specification describes the overall sequence of events or the overall motion of the system. A set of target states to be reached by the system must be defined. For more complex maneuvers, a sequence of target sets may be defined, and can be formalized by using automata from discrete event systems [20]. For example, Figures 4 and 5 show that the system must traverse the polytope $\mathcal{P}$ clockwise by moving through a sequence of triangles $\mathcal{S}_i$, $i = 1, 2, \ldots$.

The specifications above can handle complex tasks, but can be computationally difficult for high-dimensional systems; a higher-level control architecture may help to decouple the complexity and will be illustrated in our quadrocopter example.

To summarize, the control specifications require determining a polytope, which describes the allowable states and a sequence of target sets. Given this data, the objective is to find a controller that ensures the states of the robotic system remain in the polytope while reaching the correct sequence of target sets. For a related example on complex control specifications in the context of reach control, see [16].

B. Triangulation of the Polytope

To drive any initial state starting in $\mathcal{P}$ to a target state while remaining in $\mathcal{P}$ can be difficult. To systematize the design, we triangulate $\mathcal{P}$ into a set of simplices and we specify a controller on each simplex, see Figure 5. Informally, the polytope is partitioned into triangles.

Formally, an n-dimensional *simplex*, $\mathcal{S} := \text{co}\{v_0, \ldots, v_n\}$, is the convex hull of $(n+1)$ affinely independent points in $\mathbb{R}^n$; it is the generalization of a triangle. A *facet* of a simplex is a boundary face of dimension $(n-1)$. A *triangulation* is a partition of a set $\mathcal{P} \subset \mathbb{R}^n$ into n_p simplices and is denoted as $\mathbb{T} = \{\mathcal{S}_1, \ldots, \mathcal{S}_{n_p}\}$, see [19]. Then $\mathbb{T}$ satisfies the properties:

(i) $\mathbb{T} = \mathcal{S}_1 \cup \ldots \cup \mathcal{S}_{n_p}$ and

(ii) $\mathcal{S}_i \cap \mathcal{S}_j, i \neq j$, is a lower-dimensional simplex of both $\mathcal{S}_i$ and $\mathcal{S}_j$ or the empty set for all $i, j \in \{1, \ldots, n_p\}$.

Once a triangulation of $\mathcal{P}$ has been specified, the next step of the design is to identify a sequence of simplices to be visited in order to be able to reach the target states.

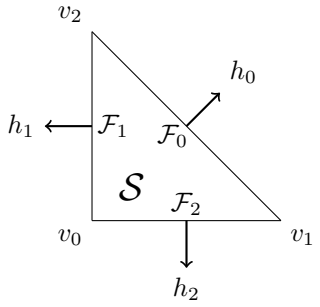

Fig. 1. Two-dimensional simplex and related terminology.

Using the sequence of simplices, *exit facets* for each simplex are designated. The trajectories starting in the given simplex may only exit the simplex through the exit facets, while the remaining facets act as *restricted facets*.

On each simplex, it is assumed that the dynamics of the system are affine, having the form

$$\dot{s} = As + Bu + a, \tag{2}$$

where $s \in \mathbb{R}^n$, $u \in \mathbb{R}^{n_u}$, and the matrices have appropriate dimensions. If the dynamics (1) are nonlinear, they may be linearized about some point in the simplex to yield the form (2).

C. Control Design via the Reach Control Problem

Finally, controllers are designed for each simplex based on the *reach control problem* (RCP). This method ensures that closed-loop trajectories flow through the designated exit facets without crossing the restricted facets. The reach control problem formulation, its theoretical developments, and conditions for solvability are discussed in [10]–[13], [15].

Below we summarize the procedure for determining a controller over an arbitrary simplex in our triangulation; see Figure 1 for a 2D example.

We define $\mathcal{I}_n = \{0, 1, \ldots, n\}$ to be the index set for the $(n+1)$ vertices of the simplex. The coordinates of each vertex are denoted as $v_i \in \mathbb{R}^n$, $i \in \mathcal{I}_n$. For each of the vertices, v_i, we must pick a corresponding $u_i \in \mathbb{R}^{n_u}$. Each facet, $\mathcal{F}_i$, of the simplex is indexed by the vertex index that it does *not* contain, and each facet has an associated normal vector h_i to describe its orientation, see Figure 1. We assume that there is at least one restricted facet, and index the restricted facets using $\mathcal{I}_r \subset \mathcal{I}_n$.

To solve RCP on a given simplex, we must select controls u_i at the vertices v_i to satisfy the so-called *invariance conditions* [12]; that is,

$$(\forall i \in \mathcal{I}_n)(\forall j \in \mathcal{I}_r \backslash \{i\}) \;\; h_j \cdot (Av_i + Bu_i + a) \leq 0. \tag{3}$$

This condition encodes that the velocity vector at each vertex points in the right direction so that trajectories leave the simplex through an exit facet while avoiding crossing the restricted facets. The feasibility of the inequalities in (3) can be easily checked numerically via a linear program. If they are not feasible, then RCP is not solvable and the choice of

restricted and exit facets must be modified. If they are feasible, then for a feasible choice of u_i, $i \in \mathcal{I}_n$, it can be shown that one can construct an affine feedback to be used over the entire simplex, see [10]. The affine feedback controller has the form

$$u(t) = K_c s(t) + g_c, \tag{4}$$

where K_c and g_c are obtained using

$$\begin{bmatrix} K_c^\top \\ g_c^\top \end{bmatrix} = \begin{bmatrix} v_0^\top & 1 \\ \vdots & \vdots \\ v_n^\top & 1 \end{bmatrix}^{-1} \begin{bmatrix} u_0^\top \\ \vdots \\ u_n^\top \end{bmatrix}. \tag{5}$$

The final step is to check that the closed-loop system, $\dot{s} = (A + BK_c)s + (a + Bg_c)$, contains no equilibrium in the simplex. If so, then RCP is solved over the simplex. For a more detailed discussion on the design of RCP controllers, see [14].

In summary, the resulting control law over $\mathcal{P}$ is a piecewise affine feedback with switching between controllers occurring at the boundaries between contiguous simplices.

III. Application to a Quadrocopter Maneuver

We follow the methodology described earlier to design a controller for executing a simple side-to-side quadrocopter maneuver. Due to the complexity of the quadrocopter system, our overall control strategy relies on a standard quadrocopter control architecture, depicted in Figure 3, that decouples the various degrees of freedom, see Section III-A. One degree of freedom, corresponding to the design of the Reach Controller in Figure 3, is responsible for controlling the side-to-side motion aspect of the maneuver and is the main point of focus in this paper. The remainder of the control architecture is standard and ensures that the quadrocopter executes the side-to-side motion while stabilizing the remaining degrees of freedom.

A. Quadrocopter Model

The quadrocopter model is ubiquitous in the literature; see, for example, [2], [3] or Chapter 4 of [4]; we refer the reader to those references for details. The vehicle dynamics are described by six degrees of freedom and are nonlinear. The translational position (x, y, z) is measured in the inertial coordinate system $\mathcal{O}$ as shown in Figure 2. The vehicle attitude is defined by the body-fixed frame $\mathcal{V}$ and is represented by the ZYX-Euler angles, yaw, pitch, and roll, (ψ, θ, ϕ). The full state of the vehicle additionally includes the translational and rotational velocities of the body frame, $(\dot{x}, \dot{y}, \dot{z})$ represented in $\mathcal{O}$ and (p, q, r) represented in $\mathcal{V}$, respectively.

In our control architecture (Figure 3), we assume that the full state of the vehicle is measured. An onboard controller takes the desired pitch angle θ_d, roll angle ϕ_d, angular body velocity around the body's z-axis r_d, and vertical velocity of the vehicle $\dot{z}_d$ as inputs and calculates the required motor forces $F_{i,d}$, $i \in \{1, 2, 3, 4\}$. In our experiment, the onboard controller is an unmodifiable blackbox. In the offboard controller, a standard, nonlinear tracking controller (as, for

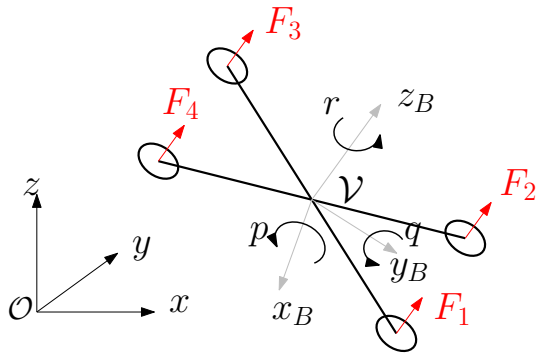

Fig. 2. The inertial and quadrocopter body-fixed frames $\mathcal{O}$ and $\mathcal{V}$. The quadrocopter is actuated by varying the thrusts F_i, $i \in \{1, 2, 3, 4\}$, produced by each motor. This results in changes to its body rotation rates, (p, q, r), and vertical acceleration, which then causes a change to the quadrocopter's position and attitude.

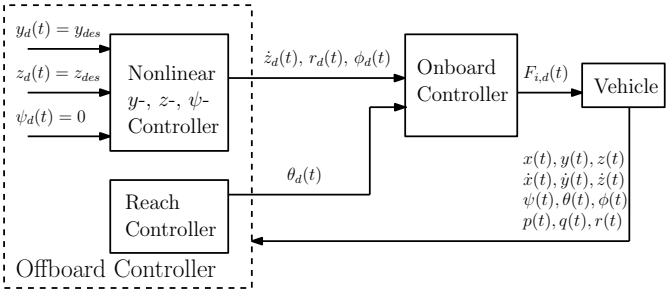

Fig. 3. The control architecture.

example, proposed in [3]) is used for stabilizing the y- and z-coordinates of the vehicle as well as the yaw. We use the Reach Controller for the x-direction.

We assume that the nonlinear controller successfully stabilizes the vehicle at $y_d(t) = y_{des}$, $z_d(t) = z_{des}$, and $\psi_d(t) = 0$, $y_{des}, z_{des} \in \mathbb{R}$, and provides the onboard controller inputs $\phi_d(t)$, $r_d(t)$, and $\dot{z}_d(t)$. The equations governing the x- and z-motion of the vehicle are then given by

$$\ddot{x}(t) = f(t) \sin(\theta(t)) \tag{6}$$
$$\ddot{z}(t) = f(t) \cos(\theta(t)) - g, \tag{7}$$

where g is the gravitational constant and $f(t)$ is the collective thrust normalized by the vehicle mass m,

$$f(t) = \frac{1}{m} \sum_{i=1}^{4} F_i(t), \tag{8}$$

with motor forces F_i, $i \in \{1, 2, 3, 4\}$, see Figures 2 and 3.

Since $z(t) = z_{des}$ implies $\ddot{z}(t) = 0$, equation (7) gives $f(t) = g/\cos(\theta(t))$, and with (6) we have

$$\ddot{x}(t) = g \tan(\theta(t)) := u(t) \;\Leftrightarrow\; \theta(t) = \tan^{-1}\left(\frac{u(t)}{g}\right). \tag{9}$$

To summarize this analysis, if we can define an $\ddot{x}$ profile, then with $u(t) = \ddot{x}(t)$ and (9) this will produce a desired pitch angle signal $\theta_d(t)$ to be input for the onboard controller (see again Figure 3). The signal $u(t)$ will be constructed via the reach control methodology outlined earlier and this will

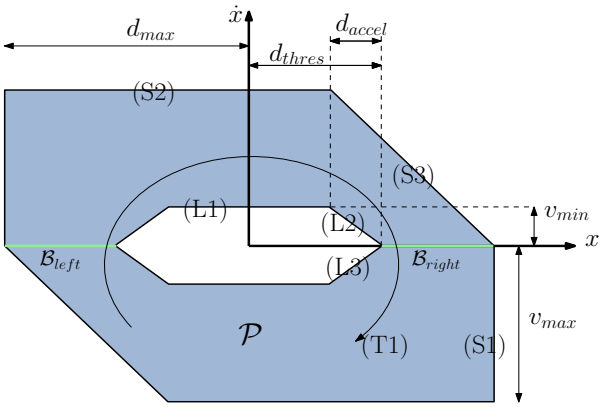

Fig. 4. Desired maneuver envelope (in blue). The specifications (S1-3), (L1-3), and (T1) are defined in Section III-B. The green lines represent the corresponding crossing sets that define when the system switches from the left-to-right to the right-to-left mode and vice versa.

yield a feedback $u(t) = u(x(t), \dot{x}(t))$.

This approach can be extended to three dimensions: similarly as above, each direction can be formulated as a single- or double-integrator, linear system. The challenge is in partitioning the state space. A recent method for complexity reduction involves working in output space [17].

B. Control Specifications

In the next step of our methodology, we state the control specifications, which result in a polytope and a sequence of target sets, see Figure 4.

a) Safety Specifications: To remain within the boundaries of the room, we require $|x| \leq d_{max}$, see Figure 4. The maximum speed limitation imposes $|\dot{x}| \leq v_{max}$. For a safe turnaround, we impose a deceleration requirement close to the room's boundaries; for compatibility with the RCP approach we use linear inequalities. We define a safe, minimum deceleration, a_{saf}, and obtain the following linear inequalities: $|x - \dot{x}/a_{saf}| \leq d_{max}$, where we choose $a_{saf} = -v_{max}/(d_{max} - d_{thres} + d_{accel}) < 0$, see Figure 4. For safety to be ensured, all three inequalities must be simultaneously satisfied at all times:

(S1) Position: $|x| \leq d_{max}$.
(S2) Speed: $|\dot{x}| \leq v_{max}$.
(S3) Deceleration: $|x - \dot{x}/a_{saf}| \leq d_{max}$.

b) Liveness Specifications: The liveness specifications ensure that the quadrocopter moves with sufficient speed when cruising between the ends of the room. Using our parameters as defined in Figure 4, we first define the inequality $|\dot{x}| \geq v_{min}$, which says the speed should be above the minimum. Next we define $|x - \dot{x}/a_{liv}| \geq d_{thres}$, and $|x + \dot{x}/a_{liv}| \geq d_{thres}$, with $a_{liv} = -v_{min}/d_{accel} < 0$. These inequalities describe how the quadrocopter should accelerate from zero speed to the minimum speed and decelerate from the minimum speed to zero speed, respectively. For liveness to be ensured, any of the three inequalities must be satisfied at all times:

(L1) Speed: $|\dot{x}| \geq v_{min}$.
(L2) Acceleration: $|x - \dot{x}/a_{liv}| \geq d_{thres}$.
(L3) Deceleration: $|x + \dot{x}/a_{liv}| \geq d_{thres}$.

c) Desired Temporal Sequence: We decompose this side-to-side maneuver into two opposing discrete modes: one in which the quadrocopter is moving from left to right, referred to as the L2R; and one in which the quadrocopter is moving from right to left, denoted by R2L. For the L2R and R2L modes respectively, we define target sets to be reached in order to transition to the opposite mode:

$$\mathcal{B}_{right} = \{(x, \dot{x}) \mid x \in [d_{thres}, d_{max}],\ \dot{x} = 0\}, \tag{10}$$
$$\mathcal{B}_{left} = \{(x, \dot{x}) \mid x \in [-d_{max}, -d_{thres}],\ \dot{x} = 0\}. \tag{11}$$

The sets are shown in Figure 4 (green lines). Assuming that the initial mode is L2R, the sequence of target sets to be crossed by the system trajectory is:

(T1) $\mathcal{B}_{right}$, $\mathcal{B}_{left}$, $\mathcal{B}_{right}$, $\mathcal{B}_{left}$, ...

For the experiment, we fix the values of the polytope parameters in Figure 4 to: $d_{max} = 2.5$ m, $d_{thres} = 1.5$ m, $d_{accel} = 0.3$ m, $v_{max} = 2$ m/s, and $v_{min} = 0.6$ m/s.

C. Triangulation and Exit Facets

Our triangulation is shown in Figure 5. The vertices of the triangulation are uniquely labeled as $\hat{v}_i$, $i \in \mathcal{I}_v := \{1, \ldots, 16\}$. The simplices are uniquely labeled as $\mathcal{S}_i$, $i \in \mathcal{I}_s := \{1, \ldots, 20\}$. Although automated procedures for triangulating and solving RCP in conjunction are considered in [18], here the triangulation was naively generated by manually partitioning $\mathcal{P}$ into simplices.

Next, to define the sequence of simplices to be visited, for each simplex we choose its restricted and exit facets. This information is also encoded in Figure 5 via the red dashed lines. For the L2R mode, the facets were chosen so as to ensure that the resulting closed-loop vector field causes trajectories to reach the set $\mathcal{B}_{right}$. Due to symmetry, the R2L can be implemented trivially using the L2R mode's design.

We remark that we also triangulated the non-liveness region (the orange region in Figure 5) in order to improve the robustness of our design. In the case that the system enters this region due to a large disturbance, the controllers defined on $\mathcal{S}_i$, $i = \{17, 18, 19, 20\}$, return the system to nominal behavior.

D. Resulting Controller Design

Now we solve RCP on each simplex $\mathcal{S}_i$, $i \in \mathcal{I}_s$. Equation (9) showed that under perfect stabilization in the y- and z-directions, the dynamics in the x-direction reduce to $\ddot{x} = u$. For this double-integrator model, the dynamics (2) over each simplex $\mathcal{S}_i$ are

$$\dot{s} = As + Bu + a = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} s + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u, \tag{12}$$

where $s := (x, \dot{x})$, whose components are the x-position and x-velocity of the quadrocopter, respectively.

With the triangulation and exit facets presented, we now construct the controllers on each simplex. For each simplex $\mathcal{S}_i$, $i \in \mathcal{I}_s$, we identify the three corresponding vertices

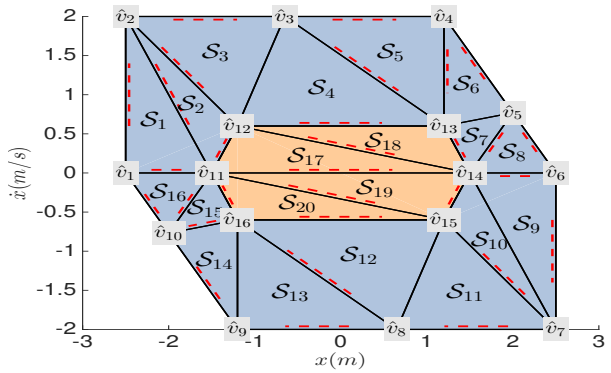

Fig. 5. The triangulation of $\mathcal{P}$, also showing exit facets, for the L2R mode. Red dashed lines represent the restricted facets of a given simplex, drawn inside of the respective simplex. The blue shaded area represents $\mathcal{P}$. The orange shaded area is guaranteeing the liveness of the system and must be avoided in nominal operating conditions.

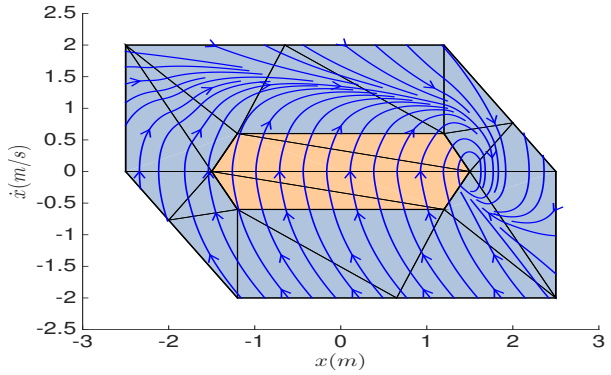

Fig. 6. The closed-loop vector field for the L2R mode, illustrated by the direction of the blue arrows.

$v_j = \hat{v}_{i_j}$, $j \in \mathcal{I}_2$, $i_j \in \mathcal{I}_v$, that form the simplex along with the corresponding exit facets $\mathcal{I}_r$. We design the control values at the three vertices such that the invariance conditions (3) are satisfied. Note that larger control values result in more aggressive controls, and hence actuator constraints can also be incorporated. Finally, we use (5) to obtain the feedback law (4). The controllers can be computed manually or using a numerical solver.

The resulting closed-loop dynamics for the L2R mode over $\mathcal{P}$ and the non-liveness region is shown in Figure 6. As expected, trajectories above the x-axis that start in $\mathcal{P}$ remain inside $\mathcal{P}$ and are guided towards the set $\mathcal{B}_{right}$, while any other non-nominal trajectory (starting in $\mathcal{P}$ below the x-axis or in the non-liveness region) eventually recovers and crosses $\mathcal{B}_{right}$. The closed-loop vector field for the R2L mode is obtained by oddly reflecting the L2R design about the origin.

We remark that to avoid discontinuities in the control when transitioning between simplices, we match the control values at the vertices along shared facets between contiguous simplices. For technical reasons related to solving RCP, it is necessary to introduce a discontinuity at $\hat{v}_{14}$ [15].

IV. Experimental Results

Our experimental platform is the Parrot AR.Drone 2.0 running firmware version 2.3.3. We interface with the AR.Drone

Fig. 7. Our quadrocopter vehicle close to the wall of the room with the motion capture camera system in the background.

through ROS, an open-source robot operating system [21]. More precisely, we used ROS Hydro, installed on a 64-bit 12.04 Ubuntu version. In addition, we used the ROS *ardrone autonomy* package [21], version 1.3.1. All experiments were conducted with the indoor hull shown in Figure 7, which protects the vehicle propellers.

We demonstrate the successful execution of the desired side-to-side motion based on our RCP approach and compare it to the performance of a standard trajectory tracking controller [2], [3], which guides the vehicle along a predefined, timed side-to-side trajectory. A video showing the experimental results can be found at: http://tiny.cc/quadrotorRCPx.

In our experiments, the following actions were performed for both trajectory tracking and the RCP approach: 1) nominal flight consisting of a few cycles of the L2R and R2L modes, 2) introducing a disturbance by manually holding the vehicle, and 3) introducing a disturbance by pushing the vehicle. At the top of Figures 8 and 9, we show the position $x(t)$ over time. The key difference that we observe is that when the quadrocopter is disturbed, the tracking approach fails the desired temporal sequence whereas the RCP approach does not. The middle plots show that the nominal behavior of both methods are comparable. The bottom plots show the significant degradation caused by the disturbances, where only in the tracking approach the trajectories exit the safety region by speeding up too much.

V. Conclusion

We succeeded in experimentally demonstrating the first ever implementation of reach controllers on a real system. The result is a logically complex quadrocopter maneuver. The main advantages of the reach control approach are that it permits the incorporation of safety constraints, event sequences and logical constraints, and robustness to unmodeled disturbances, as shown in our comparison between the reach control and standard tracking approaches. The side-to-side maneuver shown here was mainly chosen to demonstrate the proof-of-concept, and so an extension would be to apply the RCP methodology to a more complex maneuver and compare to path following controllers.

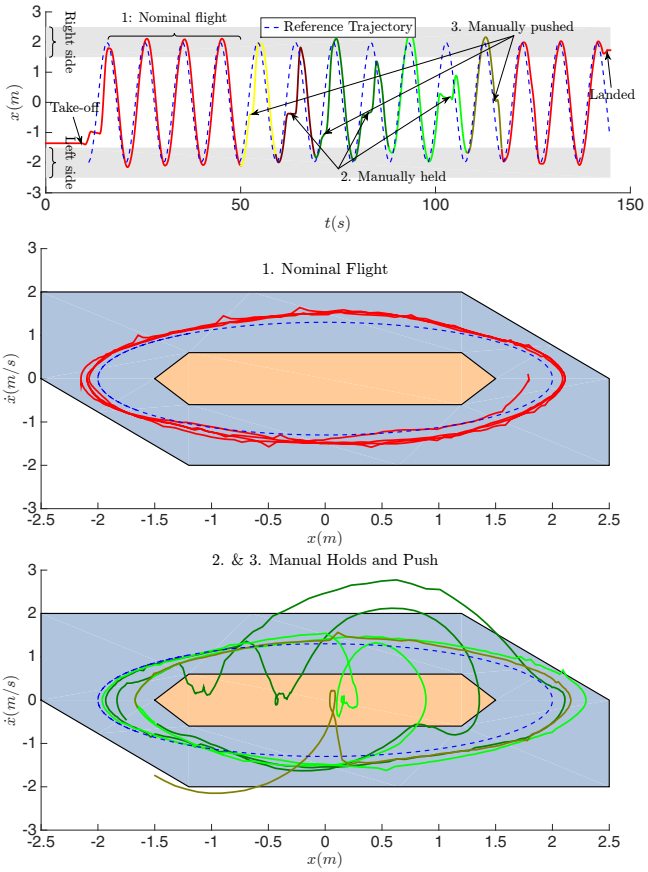

Fig. 8. Experimental results of the tracking approach.

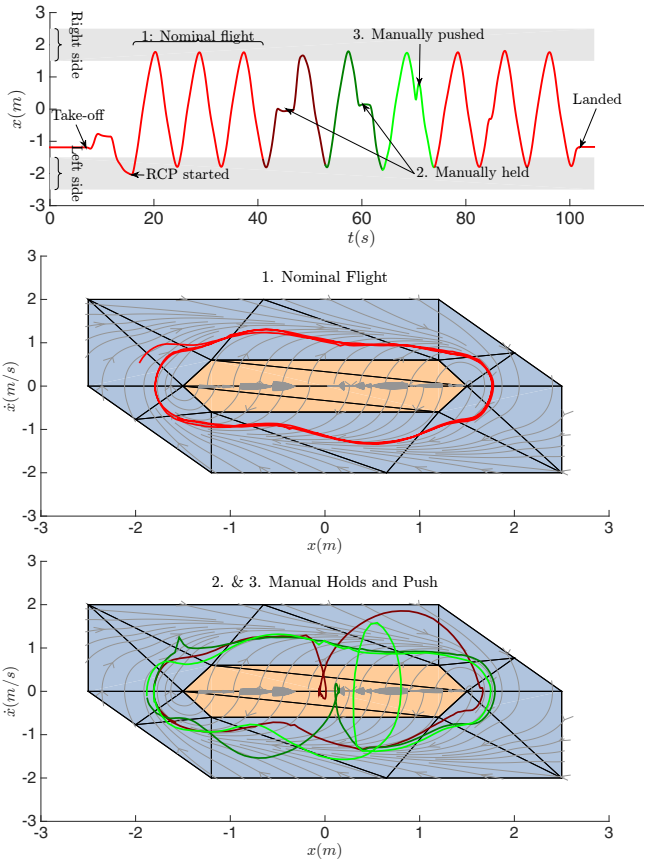

Fig. 9. Experimental results of the RCP approach.

REFERENCES

[1] A. P. Schoellig, F. Mueller, and R. D'Andrea, "Optimization-based iterative learning for precise quadrocopter trajectory tracking," *Autonomous Robots*, vol. 33, nos. 1-2, pp. 103-127, 2012.

[2] A. P. Schoellig, M. Hehn, S. Lupashin, and R. D'Andrea, "Feasibility of motion primitives for choreographed quadrocopter flight", in *Proceedings of the American Control Conference (ACC)*, 2011, pp. 3843-3849.

[3] S. Lupashin, M. Hehn, M. Mueller, A. P. Schoellig, M. Sherback, and R. D'Andrea, "A platform for aerial robotics research and demonstration: the Flying Machine Area," *Mechatronics*, vol. 24, no. 1, pp. 41-54, 2014.

[4] A. LaViers and M. Egerstedt, "Controls and art: inquires at the intersection of the subjective and the objective," Springer, 2014.

[5] D. Mellinger, N. Michael, and V. Kumar, "Trajectory generation and control for precise aggressive maneuvers with quadrotors," *The International Journal of Robotics Research*, vol. 31, no. 5, pp. 664-674, 2012.

[6] E. Xargay, I. Kaminer, A. Pascoal, N. Hovakimyan, V. Dobrokhodov, V. Cichella, A. P. Aguiar, and R. Ghabcheloo, "Time-Critical Cooperative Path Following of Multiple Unmanned Aerial Vehicles over Time-Varying Networks," *Journal of Guidance, Control, and Dynamics*, vol. 36, no. 2, pp. 499-516, 2013.

[7] A. Roza and M. Maggiore, "Path following controller for a quadrotor helicopter," in *Proceedings of the American Control Conference (ACC)*, 2012, pp. 4655 - 4660.

[8] J. Gillula, G. Hoffmann, H. Huang, M. Vitus, and C. Tomlin, "Applications of hybrid reachability analysis to robotic aerial vehicles," *The International Journal of Robotics Research*, vol. 30, no. 3, pp. 335-354, 2011.

[9] A. D. Ames, J. W. Grizzle, and P. Tabuada, "Control Barrier Function based Quadratic Programs with Application to Adaptive Cruise Control," in *Proceedings of the IEEE Conference on Decision and Control*, 2014, pp. 6271-6278.

[10] L.C.G.J.M. Habets and J.H. van Schuppen, "A control problem for affine dynamical systems on a full-dimensional polytope," *Automatica*, vol. 40, no. 1, pp. 21-35, 2004.

[11] L.C.G.J.M. Habets, P.J. Collins, and J.H. van Schuppen, "Reachability and control synthesis for piecewise-affine hybrid systems on simplices," *IEEE Transactions on Automatic Control*, vol. 51, no. 6, pp. 938-948, 2006.

[12] B. Roszak and M. E. Broucke, "Necessary and sufficient conditions for reachability on a simplex," *Automatica*, vol. 42, no. 11, pp. 1913-1918, 2006.

[13] M.E. Broucke, "Reach control on simplices by continuous state feedback," *SIAM Journal on Control and Optimization*, vol. 48, no. 5, pp. 3482-3500, 2010.

[14] G. Ashford and M.E. Broucke. Design of reach controllers on simplices. *IEEE Conference on Decision and Control*. Florence, Italy, Dec. 2013.

[15] M.E. Broucke and M. Ganness, "Reach control on simplices by piecewise affine feedback," *SIAM Journal on Control and Optimization*, vol. 52, no. 5, pp. 3261 - 3286, 2014.

[16] M. Vukosavljev and M. E. Broucke, "Control of a gantry crane: A reach control approach," in *Proceedings of the IEEE Conference on Decision and Control*, 2014, pp. 3609-3614.

[17] Z. Kroeze and M. E. Broucke, "A Viability Approach to the Output Reach Control Problem," in *Proceedings of the American Control Conference*, 2016.

[18] M. Kloetzer and C. Belta, "A fully automated framework for control of linear systems from temporal logic specifications," *IEEE Transactions on Automatic Control*, vol. 53, no. 1, pp. 287-297, 2008.

[19] C. W. Lee, "Subdivisions and triangulations of polytopes," *Handbook of Discrete and Computational Geometry*, CRC Press Series Discrete Math. Appl., pp. 271-290, 1997.

[20] R.J. Ramadge and W.M. Wonham. Supervisory control of a class of discrete event processes. *SIAM J. Control and Optimization*. Vol. 25, no. 1, pp. 206-230, 1987.

[21] Ardrone autonomy package, ver. 1.3.1, available at www.ros.org.