2012 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 7-12, 2012. Vilamoura, Algarve, Portugal

Cooperative Quadrocopter Ball Throwing and Catching

Robin Ritz, Mark W. Müller, Markus Hehn, and Raffaello D'Andrea

Abstract— **This paper presents a method for enabling a fleet of circularly arranged quadrocopters to throw and catch balls with a net. Based on a first-principles model of the net forces, nominal inputs for all involved vehicles are derived for arbitrary target trajectories of the net. Two algorithms that generate open-loop trajectories for throwing and catching a ball are also introduced. A set of throws and catches is demonstrated in the ETH Zurich Flying Machine Arena testbed.**

I. Introduction

Due to their agility and mechanical simplicity, quadrotor vehicles are an increasingly popular subject of research. A rich set of challenging tasks has been recently demonstrated. Examples include: balancing a pendulum [1]; aggressive maneuvers, such as flight through windows [2] or flips [3]; cooperative load-carrying [4]; and ball juggling with a racket [5]. Ball manipulation, such as throwing and catching, is a visually engaging problem, and without being aware of the underlying control techniques any bystander can immediately judge how successful a system is. Especially for robotic arms, a variety of successful ball catching algorithms has been demonstrated (for example in [6], [7], [8]), and recently, an approach for catching a ball with a cup mounted on a quadrocopter has been introduced [9]. In this paper, we describe a method that allows a fleet of quadrocopters, all attached to a shared net, to throw and catch balls. This touches upon various aspects of recent flying vehicle research, such as interaction between vehicles, aggressive maneuvering, real-time trajectory generation for multiple vehicles, and operating at high pitch and roll angles violating the near-hover assumption. Fig. 1 shows a snapshot of three quadrocopters attached to a shared net, which is the arrangement used to verify the feasibility of the methods introduced in this paper.

We derive a first-principles model for vehicles attached to a net in a circular formation, and based on this, generate trajectories with the intention of throwing and catching a ball. To estimate the ball's future state, we use the method described in [5]. The remainder of this paper is structured as follows: In Section II, the quadrotor dynamics and the net forces are introduced. In Section III, the quadrotor inputs for a given trajectory are derived. In Section IV, a trajectory class that minimizes the maximum acceleration is introduced, based on which we derive catching trajectories in Section V. Further, in Section VI, trajectories allowing throws are presented. Experimental results are shown in Section VII, and we conclude in Section VIII. Since we derive some equations that are too long to be shown here in an explicit form, an online appendix [10] is made available at `www.idsc.ethz.ch/people/staff/mueller-m`.

The authors are with the Institute for Dynamic Systems and Control, ETH Zürich
{rritz, mullerm, hehnm, rdandrea}@ethz.ch

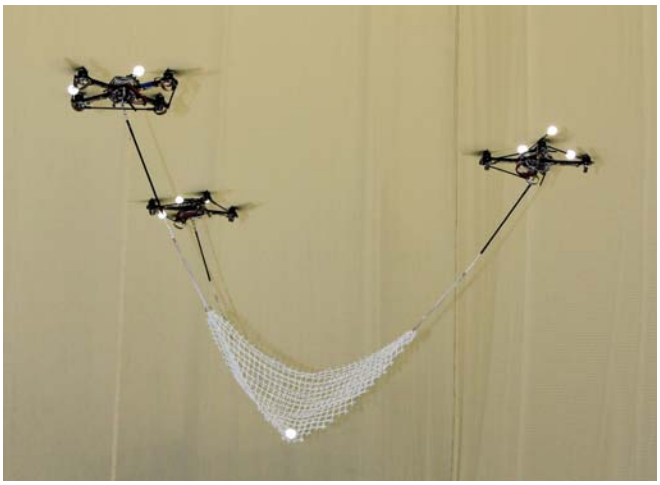

Fig. 1. Three quadrocopters attached to a net. Three retro-reflective markers are attached to each vehicle, allowing the system to determine the vehicle pose. The ball, shown lying in the net, is also retro-reflective to be visible to the motion tracking system. The vehicles are placed equally distributed on a circle around the net center.

II. Dynamics

In this section we derive the dynamics for a set of $N \geq 2$ quadrocopters that are attached in circular formation to a shared net.

A. Quadrocopter Model

Each quadrocopter $i, i \in 1, \ldots, N$ is modeled by a rigid body and therefore has six degrees of freedom: The translational position $\mathbf{p}_i = (x_i, y_i, z_i)$ is measured in the inertial coordinate frame $\mathbf{I}$, and the attitude is expressed using the zyx-Euler angles; we first rotate from the inertial frame $\mathbf{I}$ to the body-fixed frame $\mathbf{V}$ around the z-axis by yaw ψ_i, then by pitch θ_i around the current y-axis, and finally by roll ϕ_i around the current x-axis. The rotation matrix converting a vector from the body frame $\mathbf{V}$ to the inertial frame $\mathbf{I}$ yields

$$ {}^{\mathbf{V}}_{\mathbf{I}}R(\psi_i, \theta_i, \phi_i) = R_z(\psi_i) R_y(\theta_i) R_x(\phi_i), \tag{1} $$

where R_z, R_y, and R_x denote the rotation matrices about the individual axes.

1) Control Inputs: We model the quadrocopter as taking four inputs: the rotational rates about the vehicle body axes $\omega_i = (p_i, q_i, r_i)$, and the total thrust force $F_{t,i}$. We assume that the body rates ω_i can be set without dynamics and delay, because quadrocopters can reach very high angular

accelerations, and because angular rates are typically tracked by high-bandwidth on-board controllers using feedback from gyroscopes [3]. All inputs are subject to saturation, as the propeller motors and gyroscopic sensors have a limited operating range.

2) Translational Dynamics: Because the thrust force is aligned with the vehicle's z-axis, it must be converted from the body frame $\mathbf{V}$ to the inertial frame $\mathbf{I}$. Aside from thrust and gravity, an unknown force $\mathbf{F}_{net,i}$ caused by the attached net acts on the vehicle (an expression for $\mathbf{F}_{net,i}$ will be derived in Section II-B). The translational dynamics of the quadrocopter yield

$$m_{quad}\ddot{\mathbf{p}}_i = {}^{\mathbf{V}}_{\mathbf{I}}R(\psi_i,\theta_i,\phi_i)\mathbf{F}_{t,i} + \mathbf{F}_{net,i} + m_{quad}\mathbf{g}, \quad (2)$$

where m_{quad} denotes the mass of the quadrocopter, and

$$\begin{aligned}\mathbf{F}_{t,i} &= (0,0,F_{t,i}),\\ \mathbf{g} &= (0,0,-g).\end{aligned} \quad (3)$$

Notice that the mass m_{quad} is assumed to be equal for all vehicles.

3) Rotational Dynamics: To obtain the angular dynamics, the body rate inputs ω_i are converted to Euler rates. According to [1], this conversion is given by

$$\begin{bmatrix}\dot{\phi}_i\\ \dot{\theta}_i\\ \dot{\psi}_i\end{bmatrix} = \begin{bmatrix}\cos\theta_i\cos\phi_i & -\sin\phi_i & 0\\ \cos\theta_i\sin\phi_i & \cos\phi_i & 0\\ -\sin\theta_i & 0 & 1\end{bmatrix}^{-1}\begin{bmatrix}p_i\\ q_i\\ r_i\end{bmatrix}. \quad (4)$$

4) System Equations: Based on the translational and angular dynamics derived above, we define the state and the input vector for the vehicle to be:

$$\begin{aligned}\mathbf{s}_i &= (x_i, v_{x,i}, y_i, v_{y,i}, z_i, v_{z,i}, \phi_i, \theta_i, \psi_i),\\ \mathbf{u}_i &= (p_i, q_i, r_i, F_{t,i}).\end{aligned} \quad (5)$$

Finally, we can write down the first order differential equation that describes the quadrocopter's dynamics:

$$\dot{\mathbf{s}}_i = f_i(\mathbf{s}_i, \mathbf{u}_i), \quad (6)$$

where f_i combines the nonlinear equations (2) and (4).

B. Net Model

As shown in Fig. 2, the net is modeled as a point mass connected to the N vehicles by elastic strings with free-length l_{net}, where the strings are attached to the vehicle's center of gravity. The stiffness of the elastic strings is assumed to be high, such that the weight of the net itself and the weight of the ball do not lengthen the strings considerably. We assume that the vehicles are equally distributed on a horizontal circle with radius r_{net}, and we define the center of this circle to be the net position $\mathbf{p}_{net} = (x_{net}, y_{net}, z_{net})$, which is measured in the inertial frame $\mathbf{I}$. The orientation of the net is given by the yaw angle ψ_{net}; the roll and pitch angle are defined to be zero.

In accordance with the vehicles being on a circle, we define the desired position of quadrocopter $i, i \in 1,\dots,N$ to be

$$\mathbf{p}_i = \mathbf{p}_{net} + R_z(\psi_{net} + i2\pi/N)\mathbf{r}_{net}, \quad (7)$$

with $\mathbf{r}_{net} = (r_{net}, 0, 0)$. The yaw angle of vehicle i is chosen to be

$$\psi_i = \psi_{net} + i2\pi/N + \pi/2, \quad (8)$$

meaning that the y-axis of the vehicle body frame points towards the net center and the x-axis is tangent to the circle (if the vehicle's roll and pitch angle is zero). As we will see in Section VI, this choice avoids problems caused by the Euler angle singularities at $\theta_i = \pm\pi/2$ when executing throws.

To allow intuitive notations when deriving the forces caused by the net, an intermediate coordinate frame $\mathbf{F}_i$ is introduced. The intermediate frame is defined by a rotation around the inertial z-axis by a yaw angle of

$$\psi_{\mathbf{F}_i} = \psi_{net} + i2\pi/N, \quad (9)$$

hence the net center $\mathbf{p}_{net}$ and the position $\mathbf{p}_i$ of vehicle i are in an xz-plane of the intermediate frame. In Fig. 2, the relevant forces and angles to analyze the net-quadrocopter interaction are shown. The angle α is defined to be the net angle and computed as

$$\alpha = \begin{cases}\arccos\left(r_{net}/l_{net}\right) & \text{if } r_{net} < l_{net}\\ 0 & \text{otherwise}\end{cases}. \quad (10)$$

When computing the net force F_{net}, we distinguish between two cases:

Case $\alpha > 0$: The net is not stretched, and elastic forces are negligible. Assuming the inertia of the net being small compared to the inertia of the quadrocopter, we neglect the net's inertial forces. Further, for reasons of simplicity, we

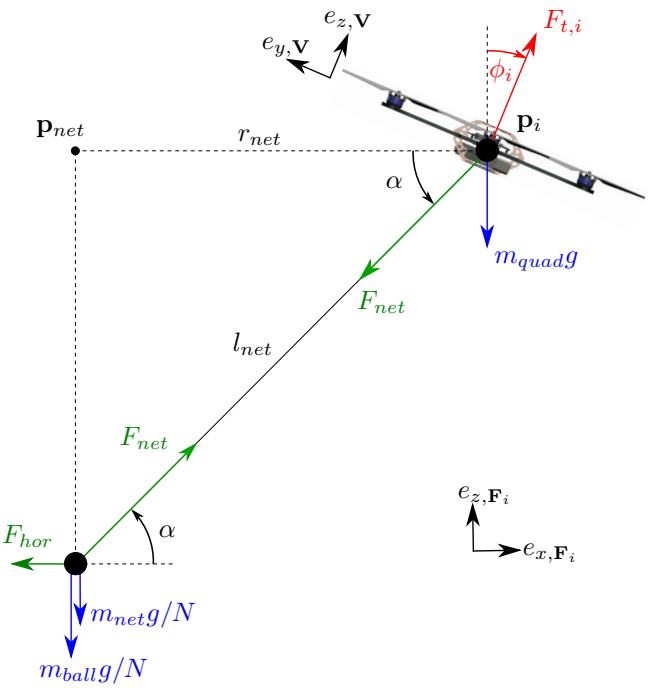

Fig. 2. Schematic illustration of the significant lengths, positions, and forces in the xz-plane of the intermediate frame $\mathbf{F}_i$. Since N vehicles are attached to the net, the masses of the ball and the net must be divided by N. For the drawing, the vehicle's pitch angle θ_i is assumed to be zero.

neglect the drag force as well; the net is stateless and only its steady-state forces are taken into account. Consequently, we can compute the net force F_{net} by balancing it against the gravitational forces of the net mass. We find

$$F_{net} = (m_{net} + m_{ball})g/(N \sin\alpha), \tag{11}$$

where m_{net} and m_{ball} denote the net and ball mass, respectively, and g is the gravitational acceleration. If the net is not currently carrying a ball, m_{ball} is set to zero. As we can see from (11), F_{net} grows to arbitrary large values if α approaches zero; this is caused by the lack of elasticity in our net model for nonzero α. At some value of F_{net}, the assumption that the strings are not being stretched is no longer justifiable. To handle this problem during a catch, the desired net radii are chosen such that the critical range of α is avoided. To handle this problem during a throw, we constrain the nominal net force to a maximum value:

$$F_{net,throw} = \min(F_{net}, F_{net,max}), \tag{12}$$

where F_{net} is computed using (11), and $F_{net,max}$ is a constant, user-defined parameter.

Case $\alpha = 0$: The net radius r_{net} exceeds the free-length l_{net} of the net strings; the elastic forces dominate due to the high string stiffness. Therefore, all other effects are neglected. The elasticity of the net is assumed to be linear, yielding the net force

$$F_{net} = k_{lin}(r_{net} - l_{net}), \tag{13}$$

where k_{lin} denotes the linear elasticity coefficient.

While the magnitude of the net force is the same for all vehicles, the direction differs. In inertial coordinates, it is given by

$$\mathbf{F}_{net,i} = R_z(\psi_{net} + i2\pi/N)\mathbf{F}_{net}, \tag{14}$$

where $\mathbf{F}_{net}$ is the net force vector in the intermediate frame:

$$\mathbf{F}_{net} = (-F_{net}\cos\alpha, 0, -F_{net}\sin\alpha). \tag{15}$$

This completes the derivation of the resulting force on vehicles attached to a net.

III. Nominal Inputs

In order to track the trajectories that will be introduced in Section V and VI, we first derive the nominal inputs for a quadrocopter attached to a net for an arbitrary net trajectory characterized by the net center, the radius, and the yaw angle trajectory. The following derivations are done in the intermediate frame $\mathbf{F}_i$; unless otherwise stated, all vectors and angles refer to the intermediate frame.

A. Thrust Force $F_{t,i}$

To derive the required thrust force $F_{t,i}$, we first compute the desired quadrocopter acceleration $\mathbf{a}_i$ based on the net trajectory by applying kinematics. Then, we insert $\mathbf{a}_i = \ddot{\mathbf{p}}_i$ into the translational dynamics (2), and solve for the thrust vector:

$$\begin{aligned}\mathbf{F}_{t,i} &= m_{quad}\mathbf{a}_i - \mathbf{F}_g - \mathbf{F}_{net,i} \\ &= \begin{bmatrix} m_{quad}a_{x,i} + F_{net}\cos\alpha \\ m_{quad}a_{y,i} \\ m_{quad}(a_{z,i}+g) + F_{net}\sin\alpha \end{bmatrix}.\end{aligned} \tag{16}$$

Consequently, the required thrust force $F_{t,i}$ is given by the magnitude of (16). It can be verified that $F_{t,i}$ increases with the net radius; at some r_{net} the desired thrust exceeds its limits and the system is no longer able to remain in steady-state.

B. Attitude $(\psi_i, \theta_i, \phi_i)$

The desired vehicle acceleration $\mathbf{a}_i$ does not only dictate the thrust input $F_{t,i}$, but the quadrocopter's attitude as well. Therefore, we seek a pitch angle θ_i and a roll angle ϕ_i that result in the desired thrust vector. As introduced in Section II, in the intermediate frame $\mathbf{F}_i$ the yaw angle ψ_i is constant at $\pi/2$. It can be verified that the resulting thrust vector of a vehicle with pitch and roll angle θ_i and ϕ_i, respectively, is then given by

$$\mathbf{F}_{t,i} = F_{t,i}(\sin\phi_i, \sin\theta_i\cos\phi_i, \cos\theta_i\cos\phi_i). \tag{17}$$

This expression for $\mathbf{F}_{t,i}$ must match the desired thrust vector given by (16). Solving the the resulting vector equation yields

$$\begin{aligned}\phi_i &= \arcsin((m_{quad}a_{x,i} + F_{net}\cos\alpha)/F_{t,i}), \\ \theta_i &= \arcsin(m_{quad}a_{y,i}/(F_{t,i}\cos\phi_i)).\end{aligned} \tag{18}$$

We note that $\phi_i > 0$ for zero acceleration, hence in steady-state, all vehicles are tilted away from the net center.

C. Body Rates (p_i, q_i, r_i)

To obtain the nominal body rate inputs (p_i, q_i, r_i), we first determine the Euler rates $(\dot{\phi}_i, \dot{\theta}_i, \dot{\psi}_i)$: By taking the total derivative of (18) with respect to time, we get expressions for $\dot{\phi}_i$ and $\dot{\theta}_i$, respectively. The vehicle's yaw rate $\dot{\psi}_i$ is equal to the net's yaw rate $\dot{\psi}_{net}$, since the yaw difference is constant. Knowing the Euler rates, the body rates can be computed by inverting (4).

IV. Minimum Maximum Acceleration Trajectory

In this section, we introduce a general, one-dimensional trajectory class that minimizes the maximum acceleration of a maneuver with fixed duration. We go on to derive catching strategies based on these trajectories in Section V.

We seek a trajectory with a given duration from a given initial to a given final state, that minimizes the absolute maximum acceleration. The coordinates (x, y, z) are decoupled by assuming that each coordinate has an independent range of allowable jerk $k_j \in [-k_{max,j}, k_{max,j}],\ j \in (x, y, z)$. The one-dimensional system dynamics for the state vector $\mathbf{s}_j = (p_j, v_j, a_j)$ are described by a triple integrator:

$$\dot{\mathbf{s}}_j = f(\mathbf{s}_j, k_j) = (\dot{p}_j, \dot{v}_j, \dot{a}_j) = (v_j, a_j, k_j). \tag{19}$$

For the remainder of the section, the subscript j indicating the coordinate is dropped to increase readability. Formally, we seek a solution to the optimizing problem

$$\begin{aligned} \text{minimize} \quad & \max\left(|a|\right) \\ \text{subject to} \quad & \dot{\mathbf{s}} = f(\mathbf{s}, k), \\ & \mathbf{s}(t_0) = \mathbf{s}_0, \\ & \mathbf{s}(t_f) = \mathbf{s}_f, \\ & k \in [-k_{max}, k_{max}] \ \forall t \in [t_0, t_f], \end{aligned} \tag{20}$$

where the boundary conditions $\mathbf{s}_0$ and $\mathbf{s}_f$, as well as the times t_0 and t_f are given. By exploiting the minimum principle [11], it can be shown that the resulting optimal maneuver has at most five intervals:

- $[t_0, t_1)$: $k = \pm k_{max}$,
- $[t_1, t_2)$: $k = 0$ and $a = \pm a_{max}$,
- $[t_2, t_3)$: $k = \mp k_{max}$,
- $[t_3, t_4)$: $k = 0$ and $a = \mp a_{max}$,
- $[t_4, t_f]$: $k = \pm k_{max}$.

A derivation of this statement is made available in [10]. Fig. 3 shows an example acceleration trajectory of such an optimal maneuver. In order to find the optimal trajectory for given initial jerk[1], we must determine the four switching times t_1, t_2, t_3, t_4, and the maximum acceleration a_{max}. For all five intervals of the trajectory, the state vector $\mathbf{s}$ can be integrated analytically, yielding polynomials in time. The constraints to be fulfilled by the solution are the three final state conditions $\mathbf{s}(t_f) = \mathbf{s}_0$, and the following condition at t_1 and t_3, respectively: Either the interval $t_2 - t_1$ vanishes, which leads to the condition $t_1 = t_2$, or $a(t_1) = \pm a_{max}$ must hold (analogous for the interval $t_4 - t_3$). Hence, five equations for five unknowns result, meaning that the solution is fully determined by the conditions. Furthermore, closed-form solutions can be found for all five unknowns and are made available in [10]. If the allowable acceleration is limited to a certain range $a \in [-a_{limit}, a_{limit}]$, then one must verify that $a_{max} \leq a_{limit}$ holds after the computation. In the limit case $a_{max} = a_{limit}$, the trajectory coincides with the trajectories derived in [12], and hence is time-optimal.

V. Catch Trajectory

In order to catch the ball, the system requires a trajectory that brings the net center to the predicted impact location before the predicted impact time. Furthermore, we decide to catch the ball, if possible, with zero net velocity and acceleration, which increases the robustness of the catching maneuver with regards to errors in the estimated impact time of the ball. Further, a large net radius at impact is beneficial for two reasons: First, assuming that the ball impacts approximately vertically, a large net radius increases the net area normal to the ball's impact direction, which, in turn, increases the robustness with respect to position errors. Second, the net force grows with the net radius, hence a large radius at impact increases the tension in the net, which reduces unpredictable net oscillations and allows for a more precise positioning of the net center. To keep the net from swinging, we want to avoid high accelerations during the catching maneuver. It is therefore a reasonable choice to construct the catch trajectory based on the minimum maximum acceleration trajectories introduced in the previous section. Further, as these trajectories can be computed quickly using the closed-form solutions, the catching trajectory can be updated regularly to account for new estimates of the ball's impact point and time.

[1]The initial jerk $k(t_0)$ is unknown, but can only take the two values $\pm k_{max}$. Hence, we attempt to compute the optimal maneuver for both, and then pick the right solution.

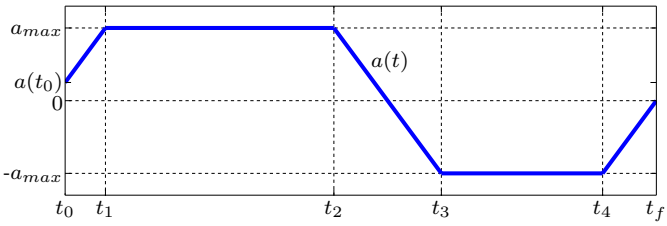

Fig. 3. Acceleration trajectory of a maneuver that minimizes $\max(|a|)$. Here, the final acceleration $a(t_f)$ is zero, which need not be the case in general.

A. Trajectory Generation

Because the maximum acceleration and jerk of the quadrocopter is limited due to actuator saturations, we must divide the control effort between going to the impact point and adjusting the net radius. We decouple these two problems and compute trajectories that minimize the maximum acceleration for each of the two subproblems. Afterwards, the acceleration trajectories are merged and their feasibility is checked. The net's yaw angle is kept constant during a catching maneuver.

1) Net Position Trajectory: For the net position trajectory, a minimum maximum acceleration trajectory is planned for each coordinate (x, y, z). The initial position of the net is obtained by averaging the positions of all attached vehicles. The desired net height $z_{net}(t_f)$ at catch is chosen by the user, and to obtain $x_{net}(t_f)$ and $y_{net}(t_f)$, we compute the crossing point in the horizontal plane in which the ball is supposed to be caught. The height of this plane is given by

$$z_{catch} = z_{net}(t_f) - l_{net} \sin \alpha(t_f), \tag{21}$$

where $\alpha(t_f)$ results from the desired net radius $r_{net}(t_f)$ at catch. The catch velocity and acceleration of the net are defined to be zero. The maximum jerk $k_{max,j}$ for each coordinate must be chosen conservatively, otherwise the desired body rates might be infeasible at the beginning of the maneuver.

2) Net Radius Trajectory: The net radius trajectory is defined by two minimum maximum acceleration trajectories: The first one enlarges the net radius before the ball is caught and ends as the ball enters the net. The second one decreases the net radius again after the catch. The corresponding maximum jerk $k_{max,r}$ for both intervals is chosen to be small in order to avoid high body rates. The distance of the radius adjustment is usually small compared to the required net center translation, hence a small $k_{max,r}$ is sufficient.

B. Feasibility

To ensure feasibility, we combine the net center and the net radius trajectory, and compute the nominal inputs as derived in Section III. If any input constraints are violated, then we iteratively reduce the desired net radius adjustment and generate a new trajectory, until we find a feasible caching maneuver. If the net radius adjustment has decreased to zero, and still no feasible solution has been found, then we enter a second iteration loop: The final velocity and acceleration are not constrained to zero anymore, but are successively increased[2]. Finally, if the impact location cannot be reached with maximum acceleration during the whole maneuver, then the catch is identified as not possible with the current settings.

VI. Throw Trajectory

We seek to plan a trajectory for the quadrocopters, such that the ball carried by the net attains a vertical velocity that results in a throw with reasonable maximum height. We intend to do purely vertical throws; the ball's nominal horizontal velocity component is zero. The vertical velocity of the ball is achieved by first accelerating vertically, and subsequently accelerating away from the net center with all vehicles. This leads to a fast rising net radius, extending the net and accelerating the ball. Since the inertia of the net is small, and since the quadrocopters pull outwards, the net suddenly and quickly decelerates when it is completely extended, and the ball, having a high vertical momentum, releases from the net and enters a free flight. After that, the vehicles are pulled back by the elastic strings, and decelerate such that they come to rest again at a desired net radius.

A. Trajectory Generation

The throw trajectory is described in the intermediate frame $\mathbf{F}_i$ where it is similar for all vehicles. The conversion to the inertial frame $\mathbf{I}$ is straightforward using (9). In the intermediate frame, the trajectory is two-dimensional; the vehicle's position y_i and yaw angle ψ_i are constant, and during a throw we define the pitch angle θ_i to be zero. For a throw taking place within the interval $t \in [t_0, t_f]$, we can describe the vehicle's trajectory by the net radius $r_{net}(t)$, the net height $z_{net}(t)$, and the roll angle $\phi_i(t) = \phi(t)$ as being equal for all vehicles. The net's yaw angle ψ_{net} is defined to be constant during a throw. For reasons of simplicity, we neglect the net force during nominal throw design, except for the interval where the net is stretched. The throw consists of four intervals, and the trajectory is generated by choosing an appropriate constant mass-normalized thrust and a constant roll rate for each of these intervals. Due to the constant inputs we can obtain the position and velocity trajectories by analytical integration[3], and consequently explicit solutions can be found for all throw parameters. Since the closed-form solutions are too large to be shown here, they are made available in [10]. Fig. 4 shows the acceleration and position trajectories of r_{net} and z_{net} for an example throw. In the following, the four throw intervals and the initial and final conditions of the throw are introduced in more detail:

Throw Start, $t = t_0$: Before a throw is started, the net must be at rest with all quadrocopters being at their steady-state configuration. The initial net radius $r_{net}(t_0)$ is a design parameter.

First Interval, $t_0 < t \leq t_1$: During the first interval, the vehicles accelerate constantly with a_{th1} along their body z-axis. Assuming that the net mass is small compared to the vehicle mass, the steady-state roll angle $\phi(t_0)$ is small as well, and the vehicles gain mostly vertical velocity during the first interval. The acceleration a_{th1} and the time t_1 when the interval ends are design parameters.

Second Interval, $t_1 < t \leq t_2$: Within this interval, the vehicles begin to turn outwards with the constant roll rate $\dot{\phi}_{th2}$. The acceleration a_{th2} is a design parameter, and the end time t_2 as well as the roll rate $\dot{\phi}_{th2}$ must be computed. To determine these two unknowns, we define the conditions

$$r_{net}(t_2) = l_{net}, \quad \phi(t_2) = \pi/2, \tag{22}$$

and hence demand that the roll angle is $\pi/2$ at the time when the net is completely extended. This is beneficial because in order to add vertical speed to the ball, we seek a high radial velocity $\dot{r}_{net}$ at the end of the second interval (which is when the ball separates from the net). If $\phi(t_2) = \pi/2$, then, towards

[2]The details of the iteration scheme are not within the scope of this paper, but the basic idea is to increase $v(t_f)$ by a constant value in each iteration, and then to compute $a(t_f)$ such that $a_{max} = a_{limit}$, hence the maneuver is time-optimal for the decoupled coordinates.

[3]Since the net force is neglected during the integration, small jumps in the desired attitude will result when we later convert the nominal position trajectory to nominal inputs. However, experimental results have shown that the low-level controller can handle these jumps and the performance is not decreased considerably.

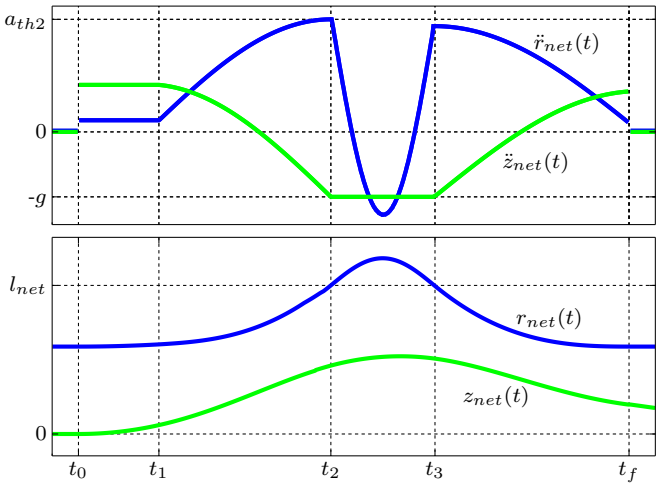

Fig. 4. Desired position and acceleration trajectories of a throw in the intermediate frame $\mathbf{F}_i$. Note that, in order to allow a more compact plot, $\ddot{r}_{net}(t)$ is scaled during the interval $[t_2, t_3]$, the actual negative acceleration is larger.

the end of the interval, we can obtain maximum horizontal acceleration for the chosen mass-normalized thrust a_{th2}.

Third Interval, $t_2 < t \leq t_3$: During the third interval, the net is being stretched and the vehicles are pulled back towards the center of the net. The movement is dominated by the net force F_{net}, given by (13). The thrust force a_{th3} is a design parameter, but its effectiveness is small. It should be chosen such that the difference to a_{th2} is not too large (to avoid large jumps in the desired propeller speed), and not near the boundaries of the allowable thrust range, such that angular corrections do not saturate the individual propeller motors. As the thrust force is constant, the net radius follows the trajectory of a linear spring-mass system with natural frequency

$$\lambda_n = \sqrt{k_{lin}/m_{quad}}. \tag{23}$$

During the whole interval, the roll angle is kept constant at $\phi = \pi/2$, since it is appropriate to enter the next interval with $\phi = \pi/2$. This is because the high negative radial velocity $\dot{r}_{net}$ must be reduced. The time t_3 when the interval is left can be computed to satisfy the condition

$$r_{net}(t_3) = l_{net}, \tag{24}$$

with $t_3 > t_2$.

Fourth Interval, $t_3 < t \leq t_f$: Within the last interval of the throw, the vehicles decelerate by applying a constant acceleration a_{th4} and a constant negative angle rate $\dot{\phi}_{th4}$. We pose the final conditions

$$r_{net}(t_f) = r_f, \;\; \dot{r}_{net}(t_f) = 0, \;\; \phi(t_f) = \phi_f, \tag{25}$$

where r_f is the desired net radius after the throw, and ϕ_f the corresponding steady-state roll angle. These three conditions determine the three unknown values a_{th4}, $\dot{\phi}_{th4}$, and t_f.

Throw End, $t = t_f$: After the throw, the horizontal velocity of the vehicles is zero, and the net has the desired radius r_f. In general, however, neither the vertical position offset $z_{net}(t_f) - z_{net}(t_0)$, nor the vertical velocity $\dot{z}_{net}(t_f)$ vanishes. This must be taken into account when planning the subsequent trajectories, e.g. to catch the ball again or to go to a desired position.

B. Feasibility

The feasibility of a throw can be verified by computing the nominal inputs based on the acceleration trajectory derived above. We allow small jumps in the desired attitude when we switch between the throw intervals, but if apart from that any input exceeds its range of allowable values, then the throw is not feasible and we have to adjust the design parameters.

C. Inclined Throws

Although the throw trajectories described above are designed for vertical throws, experimental results show that inclined throws can be achieved by tilting the resulting position trajectories. However, since we made several assumptions that are only valid for vertical throws, e.g the net force being similar for all vehicles, the tilt angle is constrained to be small. Experiments were successful until a maximum tilt angle of about 10°.

VII. Results

In this section, we show experimental results of the presented methods.

A. Experimental Setup

The experiments were carried out in the Flying Machine Arena (FMA) at ETH Zurich using modified Ascending Technologies 'Hummingbird' quadrocopters [3]. The vehicles are equipped with custom electronics, to allow the deployment of custom control algorithms. The quadrocopters receive commands at a frequency of 67 Hz through a low-latency radio link. An infrared motion-tracking system provides precise measurements of vehicle position and attitude at a rate of 200 Hz. These measurements are fed to a state observer, which delivers full state information, and compensates also for the expected closed-loop system latency. Hence, controllers can be designed for latency-free systems with full state information. For more information about the FMA we refer to [3].

B. Experimental Results

Since, especially for throws, the desired trajectories contain intervals with high tilt angles, near-hover controllers are not suitable. Therefore, the computed trajectories were tracked by linearizing the system dynamics (6) around the desired trajectory, in order to obtain a time-varying linear system which was then fed to a finite horizon time-varying LQR controller [13]. For the experiments, the throw trajectories were generated at the time when a throw was initialized, and then tracked by the controller introduced above. On the other hand, during the catches, a new trajectory was generated for each controller update with the current measured net state as initial condition and the predicted catching position as final condition. In doing so, we can account for updates of the predicted impact point whose accuracy improves as we approach the impact time. All experiments were performed with 3 vehicles, a ping-pong ball ($m_{ball} = 6$ g), and a net characterized by the parameters $l_{net} = 1.15$ m, $m_{net} = 0.12$ kg, and $k_{lin} = 107$ N/m. In Fig. (5), a series of 56 vertical throws is illustrated. The ball was always caught at the same height as it was thrown. After the catch, the net translated back to the origin before starting the next throw. Each of the 56 throws was caught successfully. Further, in Fig. (6), an isolated throw and the subsequent catch are shown; position trajectories for each coordinate are drawn for the ball, a vehicle, and the net center. We can see that, before the net is stretched, the quadrocopter's x-position has a lag compared to the desired trajectory. This is a systematic error, probably caused by the unmodeled drag and inertial forces of the net. The deviations from the target trajectory after the throw are less predictable.

VIII. Conclusion

In this paper we have presented a method for throwing and catching balls with a net attached to multiple quadrocopters. It has been shown that this aggressive, highly nonlinear task can be achieved by tracking simple trajectories, available

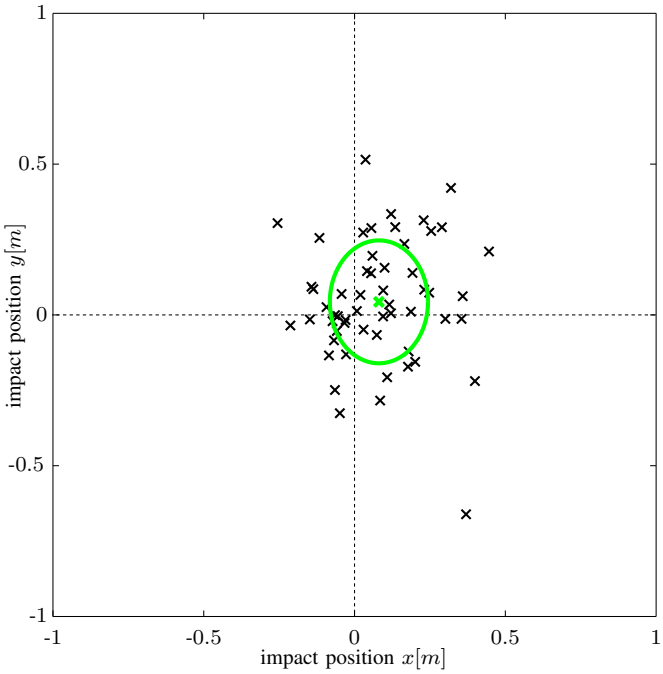

Fig. 5. Impact location for a series of 56 vertical throws with 3 vehicles. The mean values ($x_{mean} = 8.0\,\text{cm}$, $y_{mean} = 4.1\,\text{cm}$) and standard deviations ($\sigma_x = 16.3\,\text{cm}$, $\sigma_y = 20.1\,\text{cm}$) are also drawn. The height of these throws was 3.3 m on average, with a standard deviation of 10.9 cm.

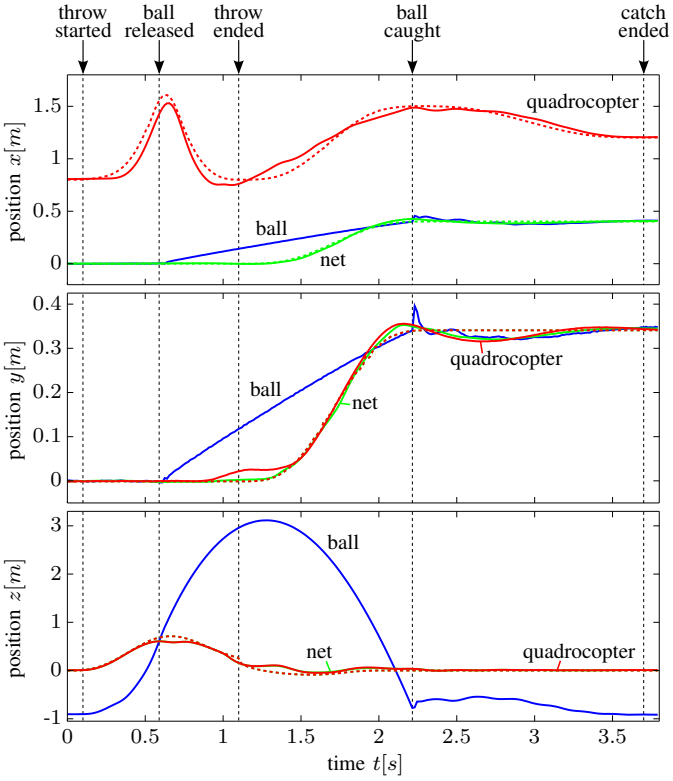

Fig. 6. Position trajectories for each coordinate of a vertical throw followed by a catch for the ball (blue), the vehicle $i = N$ (red), and the net center (green). The initial net height, as well as the desired catching height is zero, thus there is no height offset at the end of the maneuver. The solid lines denote the measured trajectories, and the dotted ones are the target positions (except for the ball that has no target trajectory). For the z-trajectory, the net and the quadrocopter trajectories are almost identical. The measured net position is obtained by averaging the three vehicle's positions, and the target net orientation is set to $\psi_{net} = 0$. We can see that, due to some disturbances, the throw is not exactly vertical, thus the net must translate to the ball's impact point. After the ball is caught, there are no longer any time constraints to be satisfied, thus the jerk to decrease the net radius again is chosen small. This is why the catching maneuver does not end shortly after the catch, but about 1.5 s later. Notice that the scaling of the position axis is not the same for the different coordinates.

as closed-form solutions. Nominal inputs for quadrocopters circularly attached to a shared mass object have been derived for an arbitrary net trajectory, which can be used for ball manipulation, but might also be useful for other applications, such as cooperative load-carrying. The feasibility of the methods presented herein has been validated in the ETH Zurich Flying Machine Arena testbed. Development potential for the future is, for example, to compensate for systematic errors during the throws by applying a learning methodology. Another future task would be to build a map between the throw parameters and the throw height, allowing the user to specify a desired height, rather than tuning the throw parameters manually. Further, an aiming algorithm could be developed, allowing throws with a substantial horizontal component towards a desired target point.

REFERENCES

[1] M. Hehn and R. D'Andrea, "A flying inverted pendulum," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, may 2011, pp. 763 –770.

[2] D. Mellinger, N. Michael, and V. Kumar, "Trajectory generation and control for precise aggressive maneuvers with quadrotors," in *Proceedings of the International Symposium on Experimental Robotics*, dec. 2010.

[3] S. Lupashin, A. Schöllig, M. Sherback, and R. D'Andrea, "A simple learning strategy for high-speed quadrocopter multi-flips," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, may 2010, pp. 1642 –1648.

[4] D. Mellinger, M. Shomin, N. Michael, and V. Kumar, "Cooperative grasping and transport using multiple quadrotors," in *Proceedings of the International Symposium on Distributed Autonomous Robotic Systems*, nov. 2010.

[5] M. Müller, S. Lupashin, and R. D'Andrea, "Quadrocopter ball juggling," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, sept. 2011, pp. 5113 –5120.

[6] B. Hove and J.-J. E. Slotine, "Experiments in robotic catching," in *American Control Conference*, june 1991, pp. 380 –386.

[7] B. Bauml, T. Wimbock, and G. Hirzinger, "Kinematically optimal catching a flying ball with a hand-arm-system," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, oct. 2010, pp. 2592 –2599.

[8] G. Batz, A. Yaqub, H. Wu, K. Kuhnlenz, D. Wollherr, and M. Buss, "Dynamic manipulation: Nonprehensile ball catching," in *Control Automation (MED), 2010 18th Mediterranean Conference on*, june 2010, pp. 365 –370.

[9] P. Bouffard, A. Aswani, and C. Tomlin, "Learning-based model predictive control on a quadrotor: Onboard implementation and experimental results," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, may 2012, pp. 279 –284.

[10] R. Ritz, M. W. Müller, M. Hehn, and R. D'Andrea, "Cooperative quadrocopter ball throwing and catching: Online appendix," oct. 2012. [Online]. Available: www.idsc.ethz.ch/people/staff/mueller-m/CooperativeQuadrocopterBallThrowingAndCatchingAppendix.pdf

[11] H. P. Geering, *Optimal Control with Engineering Applications*. Springer, 2007.

[12] M. Hehn and R. D'Andrea, "Quadrocopter trajectory generation and control," in *IFAC World Congress*, aug. 2011.

[13] D. P. Bertsekas, *Dynamic Programming and Optimal Control, Vol. I, 3rd Ed.* Athena Scientific, 2005.