Gyroscopic Obstacle Avoidance in 3-D

Marin Kobilarov
Johns Hopkins University
marin@jhu.edu

***Abstract*— The paper studies gyroscopic obstacle avoidance for autonomous vehicles. The goal is to obtain a simple controller that steers around obstacles and stabilizes to a desired state. The approach is applicable under the assumption that the obstacles are convex and that the control inputs can generate the required steering forces. This work extends the existing gyroscopic obstacle avoidance methodology to a 3-D workspace, provides a basic analysis, and studies two types of simulated vehicles: fleet of satellites and a quadcopter.**

I. Introduction

This work considers obstacle avoidance for robotic vehicles navigating through a 3-D obstacle workspace. We employ gyroscopic obstacle avoidance combined with standard stabilization control laws to handle unexpected obstacles during trajectory execution. Our approach is motivated by the fact that gyroscopic forcing has desirable convergence properties under certain conditions. In particular, the system is guaranteed to avoid collisions and reach the goal as long as the obstacle is convex and its controls are not saturated.

Obstacle avoidance is of key importance in robotics and has a long history (see [1] and [2] for an overview). Core methodologies include potential fields [3] and artificial navigation functions [4]. Potential fields are easy to implement and typically work well in simple problems but generally suffer from getting stuck in a local minima and might require careful tuning to guarantee that the vehicle reaches a goal. While navigation function overcome these limitations by providing globally-defined convergent feedback laws, they are generally computationally expensive to construct especially when dealing with complicated high-dimensional obstacle geometries. Unlike navigation functions, the dynamic window [5] is a local approach based on approximating the reachable space locally with a set of collision-free trajectories. Gyroscopic avoidance [6], employed in this work, is another strategy based on steering away from obstacles without injecting additional energy (e.g. from repulsive potential forces) into the system. By designing the system to have minimum energy at a given goal state, the system will then reach that state even in presense of obstacles. One of the limitation is that these guarantees applies only to convex obstacles. Yet, such an approach can be extremely useful in scenarios requiring fast reactive response during e.g. high-speed navigation with pop-up obstacles or formation flying.

Gyroscopic avoidance is one of the simplest steering strategies applicable to one or multiple vehicles [6], [7] that operate in a configuration space with convex obstacles. It is typically used to locally avoid an obstacle and arrive at a nearby goal state. The most common situation is when encountering unexpected obstacles while tracking a planned reference trajectory. Such type of steering for instance was employed in [8] for closed-loop control along a reference trajectory computed using other means such as differential flatness. In addition, related steering laws give rise to interesting boundary following and pattern formation behaviors in the context of interacting particles [9].

The main contribution of this work is to extend gyroscopic avoidance to vehicles modeled as fully or under-actuated rigid bodies operating in 3-D workspaces and to report on numerical studies using two non-trivial robotic models.

II. Problem Formulation

The robot is modeled as a single rigid body with position $\boldsymbol{x} = (x, y, z) \in \mathbb{R}^3$ and orientation matrix $R \in \mathrm{SO}(3)$. The *body-fixed* angular velocity is denoted by $\boldsymbol{\omega} \in \mathbb{R}^3$. The vehicle has mass m and principal moments of rotational inertia J_x, J_y, J_z forming the inertia tensor $\mathbb{J} = \mathrm{diag}(J_x, J_y, J_z)$. The state space of the vehicle is $S = SE(3) \times \mathbb{R}^6$ with $\boldsymbol{s} = ((R, \boldsymbol{x}), (\boldsymbol{\omega}, \dot{\boldsymbol{x}})) \in S$ denoting the whole state of the system.

The vehicle is actuated with control inputs $\boldsymbol{u} \in U$ where $U \subset \mathbb{R}^c$ is a bounded set. The functions $\boldsymbol{\tau} : S \times U \to \mathbb{R}^3$ and $\boldsymbol{f} : S \times U \to \mathbb{R}^3$ map from these control inputs to the resulting torques and forces acting on the body, respectively. The external forces and torques are denoted by the functions $\boldsymbol{\tau}_{\text{ext}} : S \to \mathbb{R}^3$ and $\boldsymbol{f}_{\text{ext}} : S \to \mathbb{R}^3$, respectively.

The equations of motion are

$$\dot{R} = R\hat{\boldsymbol{\omega}}, \tag{1}$$

$$\mathbb{J}\dot{\boldsymbol{\omega}} = \mathbb{J}\boldsymbol{\omega} \times \boldsymbol{\omega} + \boldsymbol{\tau}_{\text{ext}}(\boldsymbol{s}) + \boldsymbol{\tau}(\boldsymbol{s}, \boldsymbol{u}), \tag{2}$$

$$m\ddot{\boldsymbol{x}} = \boldsymbol{f}_{\text{ext}}(\boldsymbol{s}) + \boldsymbol{f}(\boldsymbol{s}, \boldsymbol{u}), \tag{3}$$

where the map $\hat{\cdot} : \mathbb{R}^3 \to \mathfrak{so}(3)$ is defined by

$$\hat{\boldsymbol{\omega}} = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}. \tag{4}$$

Obstacle constraints require that the vehicle must not collide with obstacles denoted by $\mathcal{O}_1, ..., \mathcal{O}_{n_o} \subset \mathbb{R}^3$. Assume that the vehicle is occupying a region $\mathcal{A}(R, \boldsymbol{x}) \subset \mathbb{R}^3$, and let $\mathbf{prox}(\mathcal{A}_1, \mathcal{A}_2)$ be the Euclidean distance between two sets $\mathcal{A}_{1,2}$ that is negative in the case of intersection. Obstacle avoidance requires that

$$\min_i \mathbf{prox}\left(\mathcal{A}(R, \boldsymbol{x}), \mathcal{O}_i\right) > 0. \tag{5}$$

We use a standard collision checking algorithm implemented by Proximity Query Package (PQP) [10] to compute $\mathbf{prox}$.

III. Gyroscopic Obstacle Avoidance

The goal of gyroscopic steering is to modify the dynamical system so that obstacles are avoided without violating its stability properties. This is accomplished by adding a gyroscopic force term, i.e. that does not inject energy into the system. Our development in this context can be regarded as an extension of [6] to 3-D workspaces and to vehicles with more realistic dynamics.

Assume that while executing a reference trajectory $\boldsymbol{x}_r : [0,T] \to \mathbb{R}^3$ at time $t < T$ the vehicle encounters an unexpected obstacle blocking its path. Obstacle avoidance is performed by first assigning a new desired position state $(\boldsymbol{x}_d, \dot{\boldsymbol{x}}_d)$, for instance $\boldsymbol{x}_d = \boldsymbol{x}_r(t_g)$ for some $t < t_g \leqslant T$ and then steering away from the obstacle and towards $\boldsymbol{x}_d$. This is accomplished by defining the error term

$$\Delta\boldsymbol{x} = \boldsymbol{x} - \boldsymbol{x}_d$$

and the control law

$$\boldsymbol{f}_d = -k_x\Delta\boldsymbol{x} - k_v\Delta\dot{\boldsymbol{x}} - \boldsymbol{f}_{\text{ext}} + \Gamma(\boldsymbol{s})\dot{\boldsymbol{x}}, \tag{6}$$

where $k_x, k_v > 0$ and $\Gamma(\boldsymbol{s})$ is a skew-symmetric map, i.e. such that $\Gamma = -\Gamma^T$.

Let $d(R,\boldsymbol{x}) > 0$ and $\boldsymbol{n}(R,\boldsymbol{x}) \in \mathbb{R}^3$ denote the distance and the unit vector to the detected obstacle, respectively, computed by the **prox** function. We then set

$$\Gamma(\boldsymbol{s}) = \frac{k_g v_{\max}}{d(R,\boldsymbol{x})}\hat{\boldsymbol{c}}, \tag{7}$$

for some fixed $k_g, v_{\max} > 0$, and $\boldsymbol{c} \in \mathbb{R}^3$ is computed according to

$$\begin{aligned} \boldsymbol{c} &= \begin{cases} 0 & \text{if } |\beta| \geqslant \pi/2, \\ \frac{\boldsymbol{c}'}{\|\boldsymbol{c}'\|} & \text{if } |\beta| < \pi/2, \end{cases} \\ \boldsymbol{c}' &= \frac{\boldsymbol{n}\times\dot{\boldsymbol{x}}}{\|\dot{\boldsymbol{x}}\|}, \\ \beta &= \operatorname{sgn}(\arcsin\|\boldsymbol{c}'\|)\arccos\frac{\boldsymbol{n}^T\dot{\boldsymbol{x}}}{\|\dot{\boldsymbol{x}}\|}. \end{aligned} \tag{8}$$

Here $\beta \in [-\pi, \pi]$ is angle between direction of collision and direction of motion, and $\boldsymbol{c}$ plays the role of an axis around which the velocity vector $\dot{\boldsymbol{x}}$ rotates to avoid the obstacle.

Gyroscopic avoidance modifies the dynamics but retains the stability properties of the system. To show that the desired state $(\boldsymbol{x}_d, \dot{\boldsymbol{x}}_d)$ can be reached define the function

$$V = \frac{1}{2}k_x\|\Delta\boldsymbol{x}\|^2 + \frac{1}{2}m\|\Delta\dot{\boldsymbol{x}}\|^2 \geqslant 0. \tag{9}$$

Taking its derivative and substituting the dynamics (3) and the control law (6) we obtain

$$\begin{aligned} \dot{V} &= k_x\Delta\boldsymbol{x}^T\Delta\dot{\boldsymbol{x}} + \Delta\dot{\boldsymbol{x}}^T(\boldsymbol{f}_d + \boldsymbol{f}_{\text{ext}}) && (10) \\ &= -k_v\|\Delta\dot{\boldsymbol{x}}\|^2 - \dot{\boldsymbol{x}}_d^T\Gamma(\boldsymbol{s})\dot{\boldsymbol{x}}. && (11) \end{aligned}$$

If $\dot{\boldsymbol{x}}_d$ is selected so that

$$\dot{\boldsymbol{x}}_d^T\Gamma(\boldsymbol{s})\dot{\boldsymbol{x}} \geqslant 0 \tag{12}$$

then $\dot{V} \leqslant 0$ and by LaSalle's invariance theorem the goal is asymptotically stable. The simplest way to guarantee that is to set $\dot{\boldsymbol{x}}_d = 0$ whenever obstacle-avoidance is active, i.e. when $\Gamma(\boldsymbol{s}) \neq 0$. This is equivalent to "breaking" during obstacle avoidance. More generally, satisfying (12) corresponds making sure that the desired velocity $\dot{\boldsymbol{x}}_d$ is aligned with the avoidance term $\Gamma(\boldsymbol{s})\dot{\boldsymbol{x}}$.

IV. Fully-actuated systems

Consider a class of system for which the control inputs have the form

$$\begin{bmatrix} \boldsymbol{\tau}(\boldsymbol{s},\boldsymbol{u}) \\ \boldsymbol{f}(\boldsymbol{s},\boldsymbol{u}) \end{bmatrix} = \begin{bmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & R \end{bmatrix} B\boldsymbol{u},$$

where B is a constant matrix such that $\operatorname{rank}(B) = 6$ and $\{B\boldsymbol{u} \mid \boldsymbol{u} \in U\} \subset \mathbb{R}^6$ is an open set containing the origin.

In this fully-actuated case the system can stabilize to any state specified by both position and attitude, i.e. $(\boldsymbol{x}_d, R_d) : [0,T] \to \mathbb{R}^3 \times SO(3)$ as long as the required control inputs are not saturated. It is easy to show that this can be accomplished by setting the controls to

$$\boldsymbol{u} = (B^TB)^{-1}B^T\begin{bmatrix} \boldsymbol{\tau}_d \\ R^T\boldsymbol{f}_d \end{bmatrix}, \tag{13}$$

where $\boldsymbol{f}_d$ is defined in (6) and $\boldsymbol{\tau}_d$ is a standard attitude tracking control law. For instance, following [11] such a control law is obtained by

$$\begin{aligned} \boldsymbol{\tau}_d = &-\operatorname{skew}(K_R\Delta R)^\vee - K_\omega(\boldsymbol{\omega} - \Delta R^T\boldsymbol{\omega}_d) \\ &+ \boldsymbol{\omega}\times\mathbb{J}(\Delta R^T\boldsymbol{\omega}_d) + \mathbb{J}(\Delta R^T\dot{\boldsymbol{\omega}}_d), \end{aligned} \tag{14}$$

where $\Delta R := R_d^TR$, $\operatorname{skew}(A) := (A - A^T)$, the operator $\check{\cdot}$ is the inverse of $\hat{\cdot}$ defined in (4), and K_R, K_ω are positive definite matrices.

The complete algorithms is summarized below:

Algorithm 1: $\boldsymbol{u} = \texttt{Stabilize}\,(\boldsymbol{x}_d, \dot{\boldsymbol{x}}_d, R_d, \boldsymbol{\omega}_d)$

current state: $\boldsymbol{s} = (\boldsymbol{x}, \dot{\boldsymbol{x}}, R, \boldsymbol{\omega})$
parameters: gains $k_x, k_v, K_R, K_\omega, k_g$
$\boldsymbol{f}_d = -k_x\Delta\boldsymbol{x} + k_v\Delta\dot{\boldsymbol{x}} - \boldsymbol{f}_{\text{ext}}(\boldsymbol{s}) + \Gamma(\boldsymbol{s})\dot{\boldsymbol{x}}$
$\boldsymbol{\tau}_d = -\operatorname{skew}(K_R\Delta R)^\vee - K_\omega(\boldsymbol{\omega} - \Delta R^T\boldsymbol{\omega}_d) + \boldsymbol{\omega}\times\mathbb{J}(\Delta R^T\boldsymbol{\omega}_d) + \mathbb{J}(\Delta R^T\dot{\boldsymbol{\omega}}_d)$,
$\boldsymbol{u} = (B^TB)^{-1}B^T\begin{bmatrix} \boldsymbol{\tau}_d \\ R^T\boldsymbol{f}_d \end{bmatrix}$,
return *control inputs* $\boldsymbol{u}$

V. Under-actuated systems

We next consider a class of mechanical systems modeled as underactuated rigid bodies. Assume that the system has four control inputs $\boldsymbol{u} = (u_1, ..., u_4)$ that result in torques $\boldsymbol{\tau}$ around all three axis but only a single translational force, denoted by $u \in \mathbb{R}$, along a constant body-fixed direction $\boldsymbol{e} \in \mathbb{R}^3$. Typical applications include control of aerial or surface vehicles. The control functions are such that $\boldsymbol{u}$ is in an 1-1 relationship with the torques $\boldsymbol{\tau}$ and forces $\boldsymbol{f} = Re\boldsymbol{u}$.

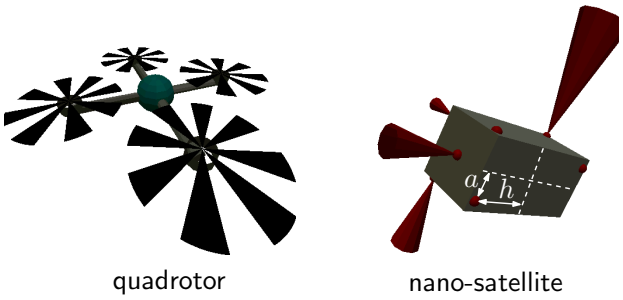

Fig. 1. Vehicle models studied.

Tracking and stabilization for an underactuated rigid body is a nontrivial problem. Only a subset of the degrees of freedom can be freely specified. It has been shown that controlling the position and the angle around the translation force input axis $\boldsymbol{e}$ can be accomplished using feedback linearization and backstepping [12], [13], [14]. We follow such an approach but focus on position stabilization control only since controlling orientation around $\boldsymbol{e}$ does not affect the obstacle avoidance control design. The complete avoidance and position stabilization algorithm is given below, with details of the derivation included in Appendix .

Algorithm 2: $(u, \boldsymbol{\tau}) = \texttt{Stabilize}\,(\boldsymbol{x}_d, \dot{\boldsymbol{x}}_d, h)$

current state: $\boldsymbol{s} = (\boldsymbol{x}, \dot{\boldsymbol{x}}, R, \boldsymbol{\omega})$
auxiliary state: variables $(\boldsymbol{f}_d, \boldsymbol{c}, u, \dot{u})$
parameters: gains $k_x, k_v, k_\eta, k_\zeta, k_g$

1 $\boldsymbol{f}_d' = -k_x \Delta\boldsymbol{x} + k_v \Delta\dot{\boldsymbol{x}} - \boldsymbol{f}_{\text{ext}}(\boldsymbol{s}) + \Gamma(\boldsymbol{s})\dot{\boldsymbol{x}}$
2 $\dot{\boldsymbol{f}}_d \approx (\boldsymbol{f}_d' - \boldsymbol{f}_d)/h$
3 $\boldsymbol{f}_d = \boldsymbol{f}_d'$
4 $\boldsymbol{\eta} = R\boldsymbol{e}u - \boldsymbol{f}_d$
5 $\boldsymbol{c}' = R^T(-k_\eta \boldsymbol{\eta} + \dot{\boldsymbol{f}}_d - \Delta\dot{\boldsymbol{x}})$
6 $\dot{\boldsymbol{c}} \approx (\boldsymbol{c}' - \boldsymbol{c})/h$
7 $\boldsymbol{c} = \boldsymbol{c}'$
8 $\boldsymbol{\zeta} = \omega \times \boldsymbol{e}u + \boldsymbol{e}\dot{u} - \boldsymbol{c}$
9 $\boldsymbol{d} = -k_\zeta \boldsymbol{\zeta} - R^T\boldsymbol{\eta} - \boldsymbol{\omega} \times \boldsymbol{e}\dot{u} + \dot{\boldsymbol{c}}$
10 $\boldsymbol{\tau} = \mathbb{J}(\boldsymbol{e} \times \boldsymbol{d}/u) - \mathbb{J}\boldsymbol{\omega} \times \boldsymbol{\omega} - \boldsymbol{\tau}_{\text{ext}}(\boldsymbol{s})$
11 $\ddot{u} = \boldsymbol{e}^T\boldsymbol{d}$
12 $\dot{u} \approx \dot{u} + h\ddot{u}$
13 $u \approx u + h\dot{u}$
14 **return** *control inputs* $(u, \boldsymbol{\tau})$

Lines (1)– (3) compute the desired force $\boldsymbol{f}_d$ and approximate its derivative as the difference between the newly computed value $\boldsymbol{f}_d'$ and the previously recorded value $\boldsymbol{f}_d$. Lines (5)– (7) update $\boldsymbol{c}$ and $\dot{\boldsymbol{c}}$ in a similar fashion. After the torques $\boldsymbol{\tau}$ and control derivatives $\ddot{u}$ are computed on lines (10) and (11), respectively, the actual control u is integrated discretely on lines (12) and (13).

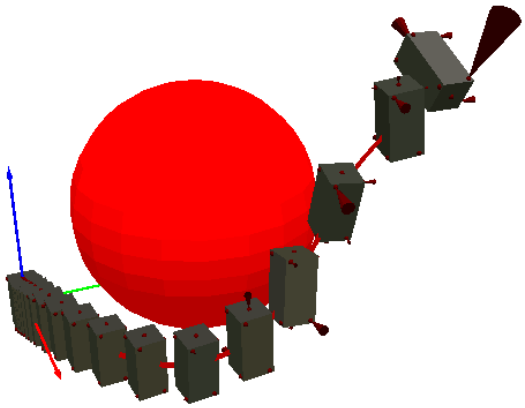

Fig. 2. A conceptual nanosatellite with ten thrusters avoiding a spherical obstacles and stabilizing at the origin.

VI. Applications

A. Satellite reconfiguration

Consider a satellite equipped with ten thrusters illustrated in Fig 1. The thruster allocation matrix is defined by

$$B = \begin{bmatrix} 0 & -h & 0 & h & h & 0 & -h & 0 & 0 & 0 \\ h & 0 & -h & 0 & 0 & h & 0 & -h & 0 & 0 \\ a & a & a & a & -a & -a & -a & -a & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 & 1 & 0 & -1 & 0 & 0 \\ 0 & -1 & 0 & 1 & -1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix},$$

where $a > 0$ and $h > 0$ are the lateral and vertical offsets of each of the side thrusters. A standard approximation of external forces is

$$\boldsymbol{f}_{\text{ext}}(\boldsymbol{s}) = m \begin{bmatrix} 2\omega_c \dot{z} \\ -\omega_c^2 y \\ -2\omega_c \dot{x} + 3\omega_c^2 z \end{bmatrix},$$

with constant $\omega_c > 0$ denoting the circular orbit angular rate. Given the time-scale considered for this simulation the effect of such forces is negligible.

Figure 2 shows an example scenario requiring the satellite to avoid a large obstacle to stabilize at the origin. Unlike the quadrotor system the fully-actuated satellite exhibited good performance in a variety of scenarios and initial conditions.

The stabilization of a group of seven satellites performing a simulated segmented mirror assembly task is considered next. The goal is to reconfigure from a free-flying mode to a latched configuration without incurring any collisions. Figure (3) shows a few snapshots of the scenario. The algorithm exhibited good behavior and no collision happened before the final latching.

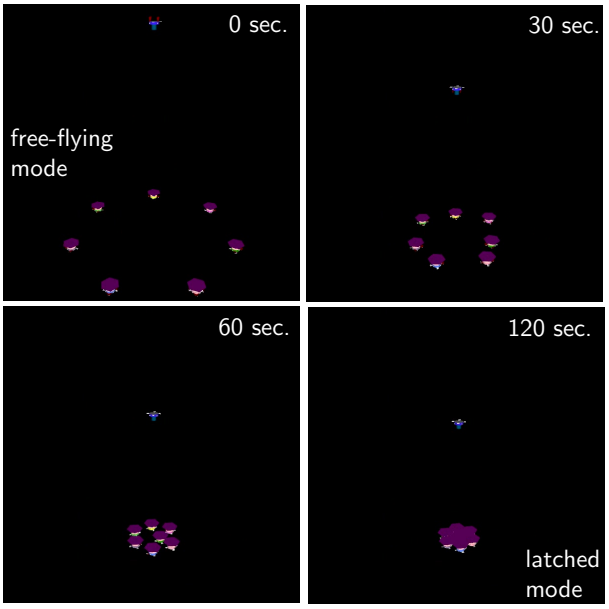

Fig. 3. Reconfiguration of a fleet of spacecraft forming a segmented mirror. Each vehicle employs gyroscopic obstacle avoidance to avoid collisions with other spacecraft and arriving stably at its latching configuration.

B. Aerial Vehicle

Consider an aerial vehicle such as a quadrotor actuated with four rotors. Typically the control functions are expressed according to

$$\boldsymbol{\tau}(\boldsymbol{s},\boldsymbol{u}) = \begin{bmatrix} lk_t(u_4^2 - u_2^2) \\ lk_t(u_3^2 - u_1^2) \\ k_m(u_1^2 - u_2^2 + u_3^2 - u_4^2) \end{bmatrix}, \qquad (15)$$

$$\boldsymbol{f}(\boldsymbol{s},\boldsymbol{u}) = R \begin{bmatrix} 0 \\ 0 \\ k_t(u_1^2 + u_2^2 + u_3^2 + u_4^2) \end{bmatrix}, \qquad (16)$$

where $u_1, .., u_4$ are the rotor speeds used as inputs and l, k_t, k_m are constants. In terms of the notation employed in §V we have

$$u = k_t(u_1^2 + u_2^2 + u_3^2 + u_4^2).$$

The external force is $\boldsymbol{f}_{\text{ext}} \approx (0, 0, -9.81m)$. Figure 4 shows an example scenario in which the quadcopter detects an obstacle when approaching within one meter and applies gyroscopic avoidance. The vehicle can safely avoid the cylinder by bending its path just enough to avoid the obstacle at low speeds. Short oscillatory behavior was observed while passing the obstacle most likely due to on-off triggering of obstacle avoidance.

VII. Conclusion

This paper studied obstacle avoidance for vehicles operating in 3-D workspaces. Our focus was on steering control laws based on gyroscopic forcing which do not alter the stability properties of the closed-loop system. Our study indicates that such steering laws lead to natural behaviors and good performance in both fully and underactuated systems.

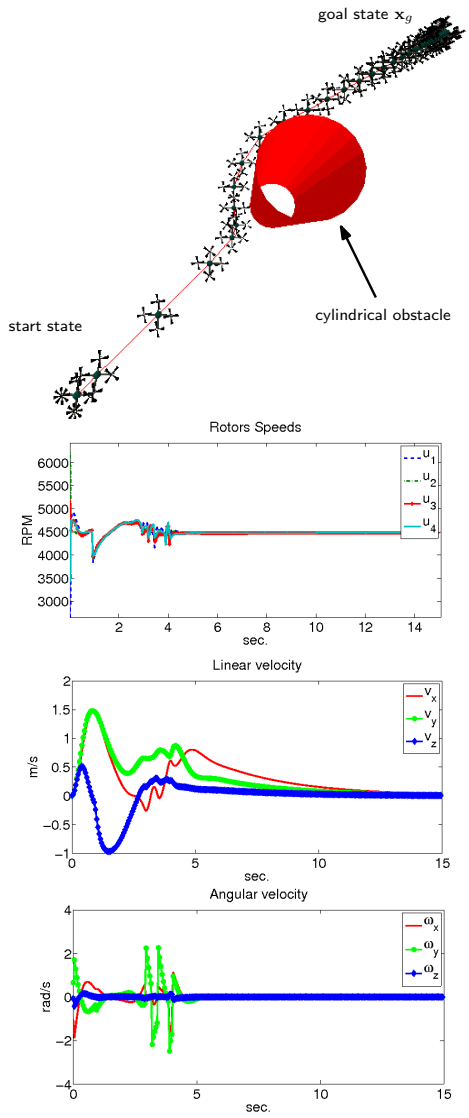

Fig. 4. Gyroscopic avoidance with a quadrotor that suddenly detects and avoids an obstacle and stabilizes at the goal position $\boldsymbol{x}_g$.

The main challenges that still need to be addressed lie in incorporating control bounds. This is a critical but non-trivial problem since due to non-trivial dynamics the exact collision states currently cannot be computed in closed-form. Studying control bounds and their effect on avoidance guarantees, as well as incorporating sensing constraints would be topics for future work.

Appendix

The controller is constructed through backstepping and satisfying nonlinear stability conditions at each step. Using the notation

$$\Delta\boldsymbol{x} := \boldsymbol{x} - \boldsymbol{x}_d.$$

define the Lyapunov function, for some $k_x, k_v > 0$,

$$V_0 = \frac{1}{2}k_x\|\Delta\boldsymbol{x}\|^2 + \frac{1}{2}m\|\Delta\dot{\boldsymbol{x}}_d\|^2 \geqslant 0. \qquad (17)$$

Differentiating and substituting the dynamics (3) gives

$$\dot{V}_0 = k_x(\Delta \boldsymbol{x})^T \Delta \dot{\boldsymbol{x}} + (\Delta \dot{\boldsymbol{x}})^T \left(\boldsymbol{f}_{\text{ext}}(\boldsymbol{s}) + R\boldsymbol{e}u - m\ddot{\boldsymbol{x}}_d\right) \tag{18}$$

Defining the *desired force* $\boldsymbol{f}_d$ by

$$\boldsymbol{f}_d = m\ddot{\boldsymbol{x}}_d - k_x \Delta \boldsymbol{x} - k_v \Delta \dot{\boldsymbol{x}} - \boldsymbol{f}_{\text{ext}}(\boldsymbol{s}) + \Gamma(\boldsymbol{s})\dot{\boldsymbol{x}}, \tag{19}$$

the relationship (18) is equivalent to

$$\dot{V}_0 = -k_v \|\Delta \dot{\boldsymbol{x}}\|^2 + (\Delta \dot{\boldsymbol{x}})^T (R\boldsymbol{e}u - \boldsymbol{f}_d + \Gamma(\boldsymbol{s})\dot{\boldsymbol{x}}). \tag{20}$$

We assume that $\dot{\boldsymbol{x}}_d$ is chosen so that $\dot{\boldsymbol{x}}_d^T \Gamma(\boldsymbol{s}) \dot{\boldsymbol{x}} \geqslant 0$. Let $\boldsymbol{\eta} = R\boldsymbol{e}u - \boldsymbol{f}_d$ and define the Lyapunov function

$$V_1 = V_0 + \frac{1}{2}\|\boldsymbol{\eta}\|^2 \geqslant 0. \tag{21}$$

Under the assumption that the following holds, for some $k_\eta > 0$,

$$\dot{\boldsymbol{\eta}} = -k_\eta \boldsymbol{\eta} - \Delta \dot{\boldsymbol{x}}, \tag{22}$$

V_1 would be a proper Lyapunov function since

$$\dot{V}_1 = \dot{V}_0 + \boldsymbol{\eta}^T(-k_\eta \boldsymbol{\eta} - \Delta \dot{\boldsymbol{x}}) = -k_v \|\Delta \dot{\boldsymbol{x}}\|^2 - k_\eta \|\boldsymbol{\eta}\|^2 \leqslant 0. \tag{23}$$

Using (1) the relationship (22) is equivalent to requiring that

$$\omega \times \boldsymbol{e}u + \boldsymbol{e}\dot{u} = \boldsymbol{c}, \tag{24}$$

where $\boldsymbol{c} \in \mathbb{R}^3$ is defined by

$$\boldsymbol{c} = R^T(-k_\eta \boldsymbol{\eta} + \dot{\boldsymbol{f}}_d - \Delta \dot{\boldsymbol{x}}). \tag{25}$$

Let $\boldsymbol{\zeta} = \omega \times \boldsymbol{e}u + \boldsymbol{e}\dot{u} - \boldsymbol{c}$ and define the Lyapunov function

$$V_2 = V_1 + \frac{1}{2}\|\boldsymbol{\zeta}\|^2 \geqslant 0. \tag{26}$$

Assuming that the following holds, for some $k_\zeta > 0$,

$$\dot{\boldsymbol{\zeta}} = -k_\zeta \boldsymbol{\zeta} - R^T \boldsymbol{\eta}, \tag{27}$$

then V_2 would be a Lyapunov function since

$$\begin{aligned} \dot{V}_2 &= \dot{V}_1 + \boldsymbol{\eta}^T R \boldsymbol{\zeta} + \boldsymbol{\zeta}^T \dot{\boldsymbol{\zeta}} \\ &= -k_v \|\Delta \dot{\boldsymbol{x}}\|^2 - k_\eta \|\boldsymbol{\eta}\|^2 - k_\zeta \|\boldsymbol{\zeta}\|^2 \leqslant 0. \end{aligned} \tag{28}$$

The relationship (27) is equivalent to the condition

$$\dot{\boldsymbol{\omega}} \times \boldsymbol{e}u + \boldsymbol{e}\ddot{u} = \boldsymbol{d}, \tag{29}$$

where

$$\boldsymbol{d} = -k_\zeta \boldsymbol{\zeta} - R^T \boldsymbol{\eta} - \boldsymbol{\omega} \times \boldsymbol{e}\dot{u} + \dot{\boldsymbol{c}}. \tag{30}$$

Condition (29) can be satisfied by setting

$$\ddot{u} = \boldsymbol{e}^T \boldsymbol{d}, \qquad \dot{\boldsymbol{\omega}} = \boldsymbol{e} \times \boldsymbol{d}/u. \tag{31}$$

Thus, using (2) the torque inputs are set to

$$\boldsymbol{\tau} = \mathbb{J}(\boldsymbol{e} \times \boldsymbol{d}/u) - \mathbb{J}\boldsymbol{\omega} \times \boldsymbol{\omega} - \boldsymbol{\tau}_{\text{ext}}(\boldsymbol{s}) \tag{32}$$

in order to render the system asymptotically stable. To summarize, the backstepping procedure was applied until the virtual input $\ddot{u}$ and the actual torques $\boldsymbol{\tau}$ were computed to satisfy the asymptotic stability requirements of the full system expressed by the Lyapunov function (26) and its negative definite derivative (28). The actual input u will in practice be computed by integrating $\ddot{u}$.

REFERENCES

[1] J.-C. Latombe, *Robot Motion Planning*. Kluwer Academic Press, 1991.

[2] H. Choset, K. M. Lynch, S. Hutchinson, G. A. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementations*. MIT Press, June 2005.

[3] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *Int. J. Rob. Res.*, vol. 5, no. 1, pp. 90–98, Apr. 1986. [Online]. Available: http://dx.doi.org/10.1177/027836498600500106

[4] E. Rimon and D. Koditschek, "Exact robot navigation using artificial potential functions," *Robotics and Automation, IEEE Transactions on*, vol. 8, no. 5, pp. 501 –518, oct 1992.

[5] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *Robotics Automation Magazine, IEEE*, vol. 4, no. 1, pp. 23 –33, mar 1997.

[6] D. E. Chang and J. E. Marsden, "Gyroscopic forces and collision avoidance with convex obstacles," *Nonlinear Dynamics and Control*, no. 295, pp. 145–160, 2003.

[7] D. Chang, S. Shadden, J. Marsden, and R. Olfati-Saber, "Collision avoidance for multiple agent systems," in *Decision and Control, 2003. Proceedings. 42nd IEEE Conference on*, vol. 1, dec. 2003, pp. 539 – 543 Vol.1.

[8] D. Vissiere, D. E. Chang, and N. Petit, "Experiments of trajectory generation and obstacle avoidance for a ugv," in *American Control Conference, 2007. ACC '07*, july 2007, pp. 2828 –2835.

[9] F. Zhang, E. Justh, and P. Krishnaprasad, "Boundary following using gyroscopic control," in *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, vol. 5, dec. 2004, pp. 5204 – 5209 Vol.5.

[10] S. Gottschalk, M. C. Lin, and D. Manocha, "OBBTree: A hierarchical structure for rapid interference detection," *Eurographics/ACM SIGGRAPH Symposium on Computer Animation*, vol. 30, pp. 171–180, 1996.

[11] F. Bullo and A. Lewis, *Geometric Control of Mechanical Systems*. Springer, 2004.

[12] T. Koo and S. Sastry, "Output tracking control design of a helicopter model based on approximate linearization," in *Decision and Control, 1998. Proceedings of the 37th IEEE Conference on*, vol. 4, dec 1998, pp. 3635 –3640 vol.4.

[13] R. Mahony and T. Hamel, "Robust trajectory tracking for a scale model autonomous helicopter," *International Journal of Robust and Nonlinear Control*, vol. 14, no. 12, pp. 1035–1059, 2004. [Online]. Available: http://dx.doi.org/10.1002/rnc.931

[14] E. Frazzoli, M. Dahleh, and E. Feron, "Trajectory tracking control design for autonomous helicopters using a backstepping algorithm," in *American Control Conference, 2000. Proceedings of the 2000*, vol. 6, 2000, pp. 4102–4107.