Autonomous Flight for Detection, Localization, and Tracking of Moving Targets With a Small Quadrotor

Justin Thomas, Jake Welde, Giuseppe Loianno, Kostas Daniilidis, and Vijay Kumar

***Abstract*—In this letter, we address the autonomous flight of a small quadrotor, enabling tracking of a moving object. The 15-cm diameter, 250-g robot relies only on onboard sensors (a single camera and an inertial measurement unit) and computers, and can detect, localize, and track moving objects. Our key contributions include the relative pose estimate of a spherical target as well as the planning algorithm, which considers the dynamics of the underactuated robot, the actuator limitations, and the field of view constraints. We show simulation and experimental results to demonstrate feasibility and performance, as well as robustness to abrupt variations in target motion.**

***Index Terms*—Aerial systems: perception and autonomy, optimization and optimal control, visual servoing.**

Fig. 1. A quadrotor accelerates to track a target moving at 1.5 m/s. The motion is planned in real time and considers the estimated trajectory of the target, the dynamics of the robot, the actuator limitations, and the camera's field of view. All sensing and computation occurs onboard the robot and leverages only one downward-facing camera and an onboard inertial measurement unit.

I. Introduction

MICRO Aerial Vehicles (MAVs) equipped with on-board sensors are becoming ideal platforms for autonomous navigation in complex, confined environments for applications such as exploration [1], inspection [2], [3], mapping [4], interaction with the environment [5], and search and rescue [6]. For truly autonomous systems, in addition to autonomous navigation, it is necessary to provide MAVs with the ability to maneuver with respect to objects, which opens the door for additional applications. Most previous works leveraging aerial robots for observation of another object assume that the object of interest is static in the world frame. However, a static object is often not a valid assumption, such as when intercepting malicious aerial vehicles, tracking moving ground-based targets, or landing on a moving vehicle. Since localization methods such as GPS do not provide information about positioning relative to an object, researchers typically consider sensors such as cameras but often overlook the limited field-of-view (FOV) constraint. The FOV constraint has been mitigated by using creative methods such as an upward-facing camera [5] or by leveraging an omnidirectional camera [7], however, aerial robots are more likely to be equipped with downward-facing sensors than upward-facing ones, and omnidirectional cameras typically require cumbersome optics, which reduces agility and payload capacity. Further, most of these works do not model the dynamics of the target or even predict its path, limiting them to quasi-static scenarios.

Thus, the goal of this work is to relax the typical assumptions of a fixed target and an unconstrained field of view, enabling a quadrotor to track a moving target while considering the robot's limited field of view. The example which will be explored is that of a robot tracking a moving sphere.

Visual odometry methods focus on determining the robot's pose relative to a starting location, not relative to a specific object, making them not directly applicable to our scenario. On the other hand, visual servoing solutions focus on motion relative to a specific object, but they typically require one or more of the following assumptions:

A1) The dynamics of the robot are first-order.
A2) The robot is fully actuated.
A3) The image features are static features on the object.
A4) The object is stationary in the inertial frame.
A5) The object does not leave the field of view.

In this work, we overcome these assumptions as they relate to maneuvering a quadrotor relative to a moving target (see, for example, Fig. 1). To accomplish this, we extend our previous planning methods for high-speed grasping and perching [8], [9].

The closest works regarding visual servoing relative to spheres develop the control for a first-order robotic arm (A1 and A2) and assume the sphere is stationary (A4) [10],

Manuscript received February 16, 2017; accepted April 14, 2017. Date of publication May 8, 2017; date of current version May 19, 2017. This letter was recommended for publication by Associate Editor E. Johnson and Editor J. Roberts upon evaluation of the reviewers' comments. This work was supported in part by the ARL under Grant W911NF-08-2-0004, in part by the ONR under Grants N00014-07-1-0829, N00014-14-1-0510, N00014-09-1-1051, and N00014-09-1-103, and in part by the NSF under Grants IIP-1113830, IIS-1426840, and IIS-1138847. *(Corresponding author: Justin Thomas.)*

The authors are with the General Robotics Automation Sensing and Perception Laboratory, University of Pennsylvania, Philadelphia, PA 19104 USA (e-mail: jut@seas.upenn.edu; jwelde@seas.upenn.edu; loiannog@seas.upenn.edu; kostas@cis.upenn.edu; kumar@seas.upenn.edu).

Digital Object Identifier 10.1109/LRA.2017.2702198

[11]. Circular markers have also been considered for visual servoing [12]. Our previous work considered servoing using a higher order underactuated system and did not require the image features to correspond to fixed locations on the target, an issue that arises when the object of interest has curvature [5]. While innovative design assisted in keeping the object in the field of view, there wasn't an explicit consideration for the field of view constraints, and there was no way to recover if the object left the image.

Our problem is also related to Proportional Navigation (PN), which was traditionally considered to control the final stages of an interceptor missle [13], [14]. However, PN-guidance systems assume that a gimballed seeker is available in the interceptor to ensure the target is continuously tracked. Further, such approaches are control strategies which do not directly address underaction and field of view constraints.

There have also been more general approaches for landing or maneuvering multirotors relative to a moving target (A4). Landing on a small carrier vehicle was accomplished by leveraging an onboard IR camera (from a Wii Remote) and IR markers on the landing pad [15]. The very limited field of view (45°) was mitigated with a pan-tilt unit. Landing on a moving target using a downward-facing camera was explored in [16], but there was no explicit consideration for the field of view. Angular dynamics were ignored but enabled landing on a vertically moving target similar to a ship deck [17]. Conveniently, this approach only relied on optical flow of the surface, making the field of view constraint less of an issue. However, the normal of the surface was assumed to be known.

The most relevant work considers tracking a moving target with a quadrotor while avoiding obstacles [18]. The trajectory of the target is estimated, and trajectories are planned to minimize the position error. In this work, we consider a similar approach with some novel extensions. Specifically, we will consider a modified objective function which is more appropriate for aggressive scenarios, and we will incorporate the field of view constraints in the robot's trajectory planner.

This work makes multiple contributions. First, we provide a solution to identify a spherical object's position with respect to the aerial platform using monocular vision. Second, a trajectory planning method enables a robot to track a moving object in real time using the single, downward-facing, body-fixed camera while explicitly considering the field of view. To the best of our knowledge, this is the first time that onboard navigation techniques based on a single camera and an inertia measurement unit (IMU) are used both to navigate in the world and track a moving target without the need of an external motion capture system or additional onboard sensors.

The high-level goal to enable a robot to track or acquire a moving target will be broken down into a number of subtasks. First, the robot must be able to determine a relative pose, which will be discussed in Section II. Since the object may be moving, we also need to estimate its motion and propagate its dynamics in order to increase robustness to occlusions and potential failure when an object may temporarily leave the field of vew. Thus, we model the object's and robot's dynamics as well as discuss the robot's controller in Section III. The planning strategy is proposed in Section IV. Finally, results are presented in Section V, and we conclude in Section VI.

II. Relative Pose

In this section, we present an approach to determine the relative pose of a sphere from a single image and knowledge of its radius. The strategy is resiliant to partial occlusions which are likely to occur during close or aggressive maneuvers when the target may not fully appear in the image. The proposed method considers fitting a cone to a set of 3D points, making the solution agnostic to the camera model and requiring only undistortion of the target's boundary in the image. Related works include fitting of ellipses and circles to points in a plane [19], [20]. Unless otherwise noted, vectors in this section will be expressed in the camera frame $\mathcal{C}$. Let a sphere be represented in the camera frame by

$$\|\mathbf{X}-\mathbf{C}\|^2 - r^2 = 0 \tag{1}$$

where $\mathbf{C}\in\mathbb{R}^3$ is the center, r is the radius, $\|\cdot\|$ is the Euclidean norm, and $\mathbf{X}\in\mathbb{R}^3$ represents any point on the surface of the sphere.

We define the projection operator π which maps a point $\mathbf{X}$ in the camera frame to a point $\mathbf{x}\in\mathbb{R}^3$ on the image surface

$$\mathbf{x} = \pi(\mathbf{X}) \equiv \frac{1}{\lambda(\mathbf{X})}\mathbf{X} \tag{2}$$

where the choice of $\lambda:\mathbb{R}^3\mapsto\mathbb{R}$ is dependent on the camera model. For example, we would choose $\lambda\equiv\|\mathbf{X}\|$ for a spherical camera model or $\lambda\equiv\mathbf{e}_3^T\mathbf{X}$, where $\mathbf{e}_3^T=\begin{bmatrix}0&0&1\end{bmatrix}$, for a pinhole model. In any case, we can express $\mathbf{X}=\lambda\mathbf{x}$, allowing us to write (1) as

$$\|\lambda\mathbf{x}-\mathbf{C}\|^2 - r^2 = 0 \tag{3}$$

which is quadratic in λ. Considering that points on the contour of the projection represent rays which are tangent to the sphere, we require λ to be unique, which means that, for the contour, the discriminant of (3) must vanish so that

$$\mathbf{x}^T\left[\mathbf{C}\mathbf{C}^T + \left(r^2-\mathbf{C}^T\mathbf{C}\right)\mathbf{I}\right]\mathbf{x} = 0, \tag{4}$$

where $\mathbf{I}$ is the identity matrix, and we observe that (4) is a conic. The set of non-trivial solutions $\mathbf{x}$ satisfying (4) and the camera model (*e.g.,* $\mathbf{e}_3^T\mathbf{x}=1$ or $\|\mathbf{x}\|=1$) describe the contour of the sphere in the image and is known as the *tangent cone* to the sphere from the camera origin [21]. In this case, however, $\mathbf{x}$ represents an observed point, making this form not ideal for the rest of our formulation where we wish to determine $\mathbf{C}$.

A. A Geometric Solution

If we assume a spherical camera model for $\mathbf{x}$ so that $\mathbf{x}^T\mathbf{x}=1$, then (4) simplifies to

$$\left(\mathbf{x}^T\mathbf{C}\right)^2 + r^2 - \mathbf{C}^T\mathbf{C} = 0. \tag{5}$$

Then, if we let $\mathbf{C}=\gamma\mathbf{c}$ with $\|\mathbf{c}\|=1$ so that $\mathbf{c}$ represents the bearing to the center of the target, we have

$$\left(\gamma\mathbf{x}^T\mathbf{c}\right)^2 + r^2 - \gamma^2 = 0, \tag{6}$$

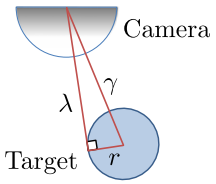

Fig. 2. The geometry for a cross section of a vector $\mathbf{X} = \lambda \mathbf{x}$ with $\|\mathbf{x}\| = 1$ in the tangent cone. The center of the sphere is given by $\mathbf{C} = \gamma \mathbf{c}$ with $\|\mathbf{c}\| = 1$. The top hemisphere represents the spherical camera model, and the circle is the cross section through a hemisphere of the sphere. Note that the Pythagorean theorem holds such that $\lambda^2 + r^2 = \gamma^2$ when a spherical camera model is used for both $\mathbf{X}$ and $\mathbf{C}$.

which is simply the Pythagorean constraint as discussed in Fig. 2. Squared errors from this constraint can be captured in a non-linear minimization problem to estimate $\mathbf{C}$

$$\arg\min_{\mathbf{C}} \sum_{i=1}^{n} \left[\left(\mathbf{x}_i^T \mathbf{C} \right)^2 + r^2 - \mathbf{C}^T \mathbf{C} \right]^2 \tag{7}$$

given n observed bearings in the tangent cone.

This minimization can be seeded with the centroid as an initial guess, and a gradient descent can be used to determine the solution. With this formulation, the geometric error is minimized, however, the computational demands may be higher than is realistic for real-time implementation.

B. An Algebraic Solution

The conic fitting problem can also be approached algebraically, analogous to algorithms used for ellipse fitting [20]. For ease of comparison, we will use similar notation.

1) Fitting a Cone to a Set of Observations: Let a conic be defined by

$$\mathbf{x}^T A \mathbf{x} = 0 \tag{8}$$

with $A \in \mathbb{R}^{3\times3}$, $A = A^T$, and $\mathbf{x} = \begin{bmatrix} x & y & z \end{bmatrix}^T$. Note that A can be arbitrarily scaled and represent the same conic. The constraints for n observations $\mathbf{x}_i$ can be written as

$$\underbrace{\begin{bmatrix} x_1^2 & y_1^2 & z_1^2 & 2x_1y_1 & 2x_1z_1 & 2y_1z_1 \\ x_2^2 & y_2^2 & z_2^2 & 2x_2y_2 & 2x_2z_2 & 2y_2z_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n^2 & y_n^2 & z_n^2 & 2x_ny_n & 2x_nz_n & 2y_nz_n \end{bmatrix}}_{D} \begin{bmatrix} A_{11} \\ A_{22} \\ A_{33} \\ A_{12} \\ A_{13} \\ A_{23} \end{bmatrix} = \mathbf{0} \tag{9}$$

so that the linear system can be solved using a singular value decomposition (SVD) of D. From this, we can construct A, which represents the best algebraic-fit conic.

2) Extracting the Relative Pose from a Conic: In a frame defined such that the z axis is aligned with the bearing to the target's center, our observations of the sphere boundary form a cone which can be written as

$$\mathbf{x}^T \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix} \mathbf{x} \equiv \mathbf{x}^T \Lambda \mathbf{x} = 0. \tag{10}$$

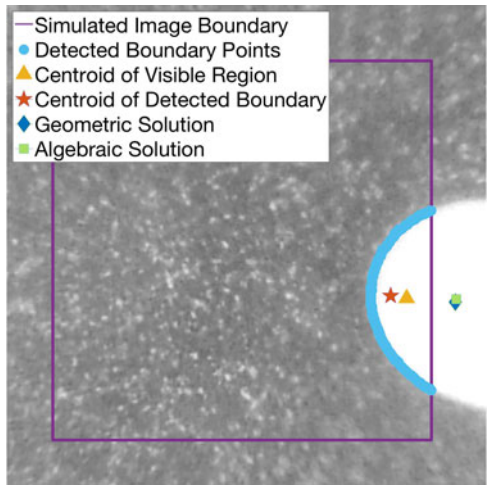

Fig. 3. A sample image of a sphere comparing various bearing estimation algorithms. The purple frame indicates a simulated image boundary. Cyan dots denote detected boundary points of the sphere. In this case, points outside the simulated image boundary are discarded to simulate a partial occlusion. The centroid of the detected region is denoted by the the yellow triangle, and the centroid of the boundary points is represented by the red pentagram. The computed bearing to the center of the sphere using the geometric solution is indicated by the blue diamond, and the computed bearing to the center of the sphere using the algebraic solution is denoted by the green square. We observe that the centroid methods do not accurately determine the bearing to the center of the sphere.

A change of basis can be applied to express the cone in camera frame coordinates with $A \in \mathbb{R}^{3\times3}$, $A = A^T$

$$\mathbf{x}^T Q \Lambda Q^{-1} \mathbf{x} \equiv \mathbf{x}^T A \mathbf{x} = 0. \tag{11}$$

Thus, we see that for a general conic defined by A, we can use an eigenvalue decomposition to determine Q, λ_1, λ_2, and λ_3. The axis of the cone in the camera frame is given by the eigenvector associated with the eigenvalue whose sign is the least common, which, without loss of generality, we can assume is λ_3. This is also the axis along which the bearing to the target's centroid will lie. For a circular cone, the other two eigenvalues would be equal and have a sign opposing λ_3. However, in practice, they will not be identical and can be approximated by their average, $\bar{\lambda}_{12}$. Then, using similar triangles, the distance to the centroid of the target can be determined to be $\gamma = r\sqrt{|\bar{\lambda}_{12}/\lambda_3| + 1}$.

C. Discussion

A comparison of these approaches is given in Fig. 3. We first note that the geometric method should provide the best fit but at the cost of the most computation time (about 23 ms). On the other hand, the algebraic approach is much faster (about 4 ms on the same machine) and provides a sufficiently accurate solution. In general, both solutions are superior to centroid methods, especially when only part of the target is observed.

If we assume a pinhole camera model, we could recover the information needed from an ellipse in the image plane. In many cases, this may be the preferred approach as there are readily available ellipse trackers such as `vpMeEllipse` from [22] and since most image processing would occur in a flat image.

With these approaches, determining the relative pose from a single image requires knowledge of the sphere's radius. A scale has been estimated online using a Structure from Motion (SfM) approach, but the relative velocity was assumed to be known [23]. In our case, the target is not stationary and the magnitude of its velocity is unknown, so determining the scale online is a more difficult problem which will be left for future work.

III. Dynamics and Control

A. Dynamics of the Object

One of the unique differences between this work and other visual-servoing works is that we no longer require the target to be fixed in the world. We simply require that the object's path can be approximated and predicted over some horizon using an n^{th} order polynomial in each Cartesian dimension.

B. Dynamics of the Robot

The dynamics of the robot are given by

$$m\ddot{\mathbf{x}} = -mg\mathbf{e}_3 + fR\mathbf{e}_3 \tag{12}$$

$$\dot{R} = R\hat{\mathbf{\Omega}} \tag{13}$$

$$\mathcal{I}\dot{\mathbf{\Omega}} = \mathbf{M} - \mathbf{\Omega} \times \mathcal{I}\mathbf{\Omega} \tag{14}$$

where m is the mass of the robot, $\mathbf{x} \in \mathbb{R}^3$ is the position, g is gravity, $\mathbf{e}_3$ is the 3rd standard basis vector, $f \in \mathbb{R}$ and $\mathbf{M} \in \mathbb{R}^3$ are the thrust and moment control inputs to the system, $R \in SO(3)$ is the orientation of the vehicle, and $\hat{\cdot} : \mathbb{R}^3 \mapsto \mathfrak{so}(3)$ is the hat map defined such that, for any two vectors, $\hat{\mathbf{a}}\mathbf{b} = \mathbf{a} \times \mathbf{b}$. Also, $\mathcal{I}$ is the inertial matrix, and $\mathbf{\Omega}$ is the robot's angular velocity expressed in the robot frame.

C. Control

We leverage a position-based visual servoing (PBVS) control strategy, which allows the use of a common nonlinear controller [24], [25]. The position and velocity errors are

$$\mathbf{e}_x = \mathbf{x} - \mathbf{x}_{\text{des}} \quad \text{and} \quad \dot{\mathbf{e}}_x = \dot{\mathbf{x}} - \dot{\mathbf{x}}_{\text{des}}, \tag{15}$$

respectively, and the thrust is computed as

$$f = (-k_x\mathbf{e}_x - k_v\dot{\mathbf{e}}_x + mg\mathbf{e}_3 + m\ddot{\mathbf{x}}_{\text{des}}) \cdot R\mathbf{e}_3 \tag{16}$$

where k_x and k_v are positive gains and the subscript "des" denotes a desired value. The attitude and angular velocity errors are defined as

$$\mathbf{e}_R = \frac{1}{2}\left(R_{\text{des}}^T R - R^T R_{\text{des}}\right)^\vee, \quad \mathbf{e}_\Omega = \mathbf{\Omega} - R^T R_{\text{des}}\mathbf{\Omega}_{\text{des}} \tag{17}$$

where $\cdot^\vee : \mathfrak{so}(3) \mapsto \mathbb{R}^3$ is the opposite of the hat map. The control moments are computed as

$$\mathbf{M} = -k_R\mathbf{e}_R - k_\Omega\mathbf{e}_\Omega + \mathbf{\Omega} \times \mathcal{I}\mathbf{\Omega}, \tag{18}$$

where k_R and k_Ω are positive gains. Then, the zero-equillibrium is exponentially stable and, in general, the controller provides "almost global exponential attractiveness" [25].

IV. Planning

Since we are interested in aggressive maneuvers so that the quadrotor can commence tracking a quickly moving target, it is important to not only consider dynamic feasiblity (considering the relative degree of the robot), but also to ensure that actuator and sensor constraints, including the field of view, are not violated. Actuator and gyro constraints were considered when planning, enabling a robot to perform aggressive maneuvers to perch on vertical surfaces using feedback from a motion capture system [9], [26]. An extension to plan trajectories for image features was presented, but there were no guarantees that the trajectories would satisfy the sensor and actuator constraints [5]. The approach here will allow for the consideration of the dynamic, sensor, and actuator constraints, including the field of view.

A. Representation of Trajectories

We express trajectories using an nth order polynomial basis with terms $b_k(t)$ so a trajectory $p(t)$ can be represented by

$$p(t) = \sum_{k=0}^{n} c_k b_k(t) \tag{19}$$

or with a vector of coefficients $\mathbf{c}_i \in \mathbb{R}^{n+1}$ for dimension i and a basis vector $\mathbf{b}(t) : \mathbb{R} \mapsto \mathbb{R}^{n+1}$

$$p_i(t) = \mathbf{c}_i^T \mathbf{b}(t). \tag{20}$$

We could allow $\mathbf{b}(t)$ to be a standard power basis,

$$\mathbf{b}(t) = \begin{bmatrix} 1 & t & t^2 & \cdots & t^n \end{bmatrix}^T, \tag{21}$$

a Legendre Polynomial basis,

$$\mathbf{b}(t) = \begin{bmatrix} 1 & t & \frac{1}{2}\left(3t^2 - 1\right) & \frac{1}{2}\left(5t^3 - 3t\right) & \cdots \end{bmatrix}^T, \tag{22}$$

or any basis of the user's choice. The rth derivative can be computed as

$$p_i^{(r)}(t) = \mathbf{c}_i^T \mathbf{b}^{(r)}(t) \tag{23}$$

since $\mathbf{c}_i$ is independent of time.

Now, let $B(t) : \mathbb{R} \mapsto \mathbb{R}^{d(n+1)\times d}$ and $\mathbf{c} \in \mathbb{R}^{d(n+1)}$ be defined as

$$B(t) = \begin{bmatrix} \mathbf{b}(t) & & \\ & \ddots & \\ & & \mathbf{b}(t) \end{bmatrix}, \quad \mathbf{c} = \begin{bmatrix} \mathbf{c}_1 \\ \vdots \\ \mathbf{c}_d \end{bmatrix} \tag{24}$$

for d dimensions. That is, $\mathbf{c}$ is a stack of the coefficient vectors. Then, we can write the trajectory as $\mathbf{p}^{(r)}(t) : \mathbb{R} \mapsto \mathbb{R}^d$ where

$$\mathbf{p}^{(r)}(t) = \left(B^{(r)}(t)\right)^T \mathbf{c}. \tag{25}$$

For clarity, we note that this is equivalent to

$$\mathbf{p}^{(r)}(t) = \begin{bmatrix} \mathbf{c}_1^T \\ \mathbf{c}_2^T \\ \vdots \\ \mathbf{c}_d^T \end{bmatrix} \mathbf{b}^{(r)}(t), \tag{26}$$

however, the previous formulation will be useful later.

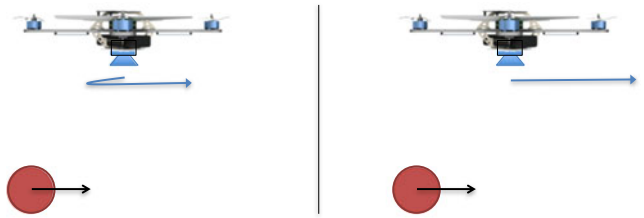

Fig. 4. A one-dimensional example motivating the minimization of velocity error between the robot and the target. In both scenarios pictured above, the robot is assumed to start from rest as the object enters the field of view. On the left hand side, we see a possible trajectory if the position error is minimized, and on the right-hand side, we see a result if the velocity error is minimized. In the position-error case (LHS), it is obvious that the motion is not ideal for larger target velocities. Thus, we are motivated, at least during the initial transient, to minimize the velocity error. Observe that this also could aid in mitigating the field of view constraints.

B. Trajectory of the Target and Robot

The estimated trajectory of the target in dimension i is defined by coefficients $\mathbf{h}_i \in \mathbb{R}^{n+1}$ so that the trajectory for all dimensions is

$$\mathbf{g}^{(r)}(t) = \left(B^{(r)}(t)\right)^T \mathbf{h} \tag{27}$$

where $\mathbf{h} = \begin{bmatrix}\mathbf{h}_1^T & \dots & \mathbf{h}_d^T\end{bmatrix}^T$. This allows us to fit a polynomial to the dynamic model over some horizon to predict the motion of the target [18]. Next, we define the planned trajectory of the robot using the convention in (25). The trajectories are specified in the space of *flat outputs* of the vehicle using the three Cartesian coordinates x, y, and z in an inertial frame. Then, dynamic feasibility is guaranteed simply by enforcing inter-segment continunity on position, velocity, acceleration, jerk ($r = 3$), and snap ($r = 4$) in each dimension [24], [26].

C. General Planning Strategy

One strategy is to minimize the position error between the target and the robot's trajectories with the inclusion of smoothing on higher derivatives [18]. However, this approach is not ideal in some scenarios (see discussion in Fig. 4) because it can produce initial transients that are counterproductive to achieving tracking as quickly as possible.

Instead, we propose minimizing the velocity error during the initial transient. Interestingly, this results in a strategy similar to the ones taken by dragonflies [27], falcons [28], and human outfielders [29]. In these cases, the target is regulated to remain at a constant bearing in the field of view, and the range gap is closed using a strategy that may be captured by τ ("tau") theory [30], which doesn't require knowledge of a distance to the target. For now, we assume that the scale of our target is known a priori, allowing for the direct estimation of the range, and we leave the online range estimation, if necessary at all, for future work.

Minimizing velocity error alone, however, cannot capture the desired relative pose. Thus, we consider the planning and tracking problem as having two phases. First, there must be a phase where the robot is accelerating to match the velocity of the target. This transient phase is when the field of view constraints are most likely to be active constraints. The objective of this phase can be expressed as a minimization of the velocity error between the robot and the target. The second phase incorporates the position error and enables planning to intercept the target or track the target from a desired relative pose.

D. A Multi-Objective Cost Function

We are motivated to use a multi-objective cost function to penalize both velocity errors and, when appropriate, position errors. Interestingly, a similar approach was used to smoothly change formation shapes of an aerial robot team [31]. We define the error as

$$\mathbf{e}(t) = \mathbf{g}(t) - \mathbf{p}(t) \tag{28}$$

over the x, y, and z coordinates of the appropriate frame such that $\mathbf{g}(t), \mathbf{p}(t), \mathbf{e}(t) : \mathbb{R} \mapsto \mathbb{R}^3$. Since we're interested in minimizing specific derivatives, we write a general objective function which computes the integrated square of the Euclidean error of the rth derivative

$$\mathcal{J}_r = \int_{t_o}^{t_f} \left\|\mathbf{e}^{(r)}(t)\right\|^2 dt \tag{29}$$

where $\|\cdot\|$ represents the Euclidean norm. Expanding, we have

$$\mathcal{J}_r = \int_{t_o}^{t_f} \left(\mathbf{e}^{(r)}\right)^T \left(\mathbf{e}^{(r)}\right) dt \tag{30}$$

$$= \mathbf{c}^T Q_r\, \mathbf{c} - 2\mathbf{h}^T Q_r\, \mathbf{c} + \mathbf{h}^T Q_r\, \mathbf{h} \tag{31}$$

where

$$Q_r = \int_{t_o}^{t_f} \left(B^{(r)}(t)\right)\left(B^{(r)}(t)\right)^T dt, \tag{32}$$

so that $\mathcal{J}_r$ can be expressed in quadratic form as

$$\mathcal{J}_r = \mathbf{c}^T Q_r \mathbf{c} + \mathbf{f}^T \mathbf{c} + \alpha,\ \mathbf{f} = -2Q_r^T \mathbf{h},\ \alpha = \mathbf{h}^T Q_r \mathbf{h}. \tag{33}$$

Note that a translation could be included in the object's coefficients, $\mathbf{h}$, to specify a desired relative pose. Further, we could nondimensionalize t so that $t_o = 0$ and $t_f = 1$, which would allow precomputation of Q_r for each derivative.

E. Actuator and Sensor Constraints

The field of view of a lens could be modeled as a cone in the camera (or body) frame

$$\mathbf{m}^T A \mathbf{m} \leq 0 \tag{34}$$

where $A = A^T$ and the solutions $\mathbf{m} \in \mathbb{R}^3$ are rays lying within the field of view. Unfortunately, A is not positive semidefinite, which means that we can not include it as a quadratic constraint in a Quadratically Constrained Quadratic Program (QCQP). Alternatively, we could model the constraint with an inscribed pyramid similar to the approximation of a coulomb friction cone (see Fig. 5 or [32]). Further, because of the rectangular sensor design of most cameras, the cone model may not be best. Thus, we can inscribe a convex pyramid in the field of view, which provides a set of linear constraints representing the effective field of view.

Fig. 5. The field of view of a lens. The cone's representation is not positive semidefinite, which means we cannot use it directly in a QCQP. However, the cone can be approximated with an inscribed pyramid.

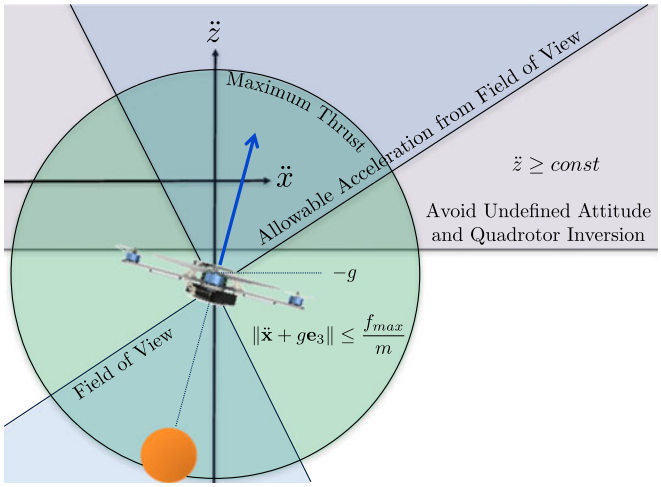

Fig. 6. A visualization of the acceleration constraints in the $x - z$ plane. The large circle represents the maximum thrust bound, and is similar to the illustration in [33]. The half plane bound keeps the quadrotor from inverting and avoids the singularity resulting when there is no thrust. Given the expected bearing between the target and the robot (inclined, dashed line), a conic (or pyramid) section represents the acceleration constraint to keep the object in the field of view.

Next, we incorporate bounds to prevent actuator saturation using the approach in [9]. However, we also want to consider the field of view constraints so the object does not leave the image. The simplest way to solve this problem is to prescribe a maximum attitude angle (*e.g.,* $\arccos(\mathbf{e}_3 \cdot R\mathbf{e}_3) \leq \beta_{\max}$), and reduce the effective field of view accordingly. A trajectory could then be planned simultaneously using the maximum attitude constraint and the reduced field of view to constrain the relative positions. However, this approach is more conservative than desired, especially when aggressive maneuvers are necessary. Certainly, we do not want to restrict the maximum attitude.

Instead, we directly incorporate the field of view as constraints in the optimization by using the relative position to rotate the field of view and prescribe constraints on the acceleration (effectively on the attitude). We first determine a set of acceleration constraints when the object would be directly beneath the robot. Then, we rotate the axis to be coincident with the expected bearing from the object to the robot. A schematic of sample acceleration constraints at one instant is given in Fig. 6. It is possible that the vision constraints can dominate the solution causing the robot to move away from the target. In such cases, the vision constraints could be relaxed, allowing the target to temporarily leave the field of view.

Algorithm 1: The Planning Algorithm.

$\mathcal{J} \leftarrow \lambda_1\mathcal{J}_1 + \lambda_3\mathcal{J}_3$
for Each Horizon **do**
  update($\mathbf{g}(t)$)
  repeat
    $\mathbf{p}(t) \leftarrow$ iterateSQP($\mathcal{J}$, $\mathbf{g}(t)$, $\mathbf{p}(t)$)
  until Out of Time
  $\mathbf{e}(t) \leftarrow \mathbf{g}(t) - \mathbf{p}(t)$
  if ($\mathbf{e} \cdot \dot{\mathbf{e}} \geq 0$ or $\|\dot{\mathbf{e}}\| \leq thresh$) **then**
    $\mathcal{J} \leftarrow \lambda_0\mathcal{J}_0 + \lambda_1\mathcal{J}_1 + \lambda_3\mathcal{J}_3$
  end if
end for

F. The Planner

In this subsection, we describe the proposed planning algorithm. We leverage a receding horizon planning strategy to continuously update the planned trajectory based on the target's actions. We first set the default objective function to minimize the velocity error, and we include a slight weighting to penalize jerk, which helps to reduce the angular velocity and has been used previously to provide smoothing [18]. Note that we could also penalize the next derivative, snap, which would most directly help to reduce the angular acceleration. Dynamic, actuator, gyro, and vision constraints are incorporated as nonlinear constraints on the coefficients of the trajectory. Then, the problem can be solved using a Sequential Quadratic Program (SQP) solver (*e.g.,* [34]). In our case, there is the benefit that instead of the QP subproblem being an approximation of a general nonlinear cost function, it is identical to our cost function. Most convergence arguments for SQP problems require that an initial solution or "warm start" is sufficiently close to the actual solution and that the active inequality constraints at the optimal solution are the same ones that are active at the local solution. For more details, we refer the reader to the "Sequential Quadratic Programming Methods" chapter in [35].

When the relative bearing and relative velocity have a positive inner product, then there is no harm in also minimizing position error (see the example in Fig. 4). Additionally, there may be situations where the velocity is matched before the previous condition is satisfied. In such cases, we can also incorporate the position term in the objective function once the relative velocity falls below a predefined threshold. While this planning approach does not guarantee completeness, it is viable for real-time applications, and it works well in practice. The algorithm is presented in Algorithm 1, where the weighting for derivative r is given by λ_r, and $\mathcal{J}_r$ is defined by (33).

V. Results

In this section, we present our simulation and experimental results. We first present a sample simulation assuming a

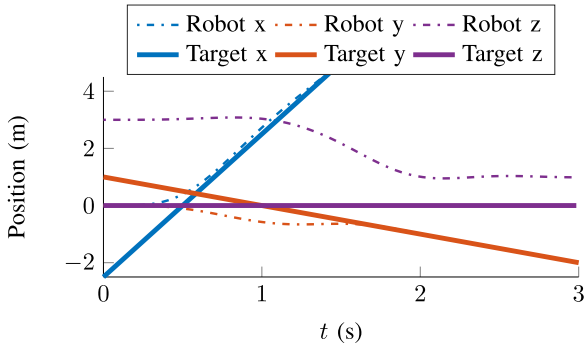

Fig. 7. The nominal positions from the proposed planning strategy.

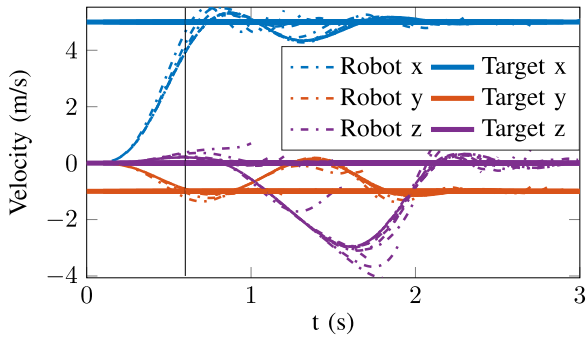

Fig. 8. The planned velocities resulting from the receding horizion planner. All planned trajectories are shown as dash-dotted lines, but only the first 0.1 s of any trajectory is executed before the next one is planned. The planning horizon is 1 s. We observe interesting results like the initial positive velocity in z, which is possibly a result of the field of view constraint. At $t = 0.6$ s (represented by the vertical line), the position error is included in the cost function.

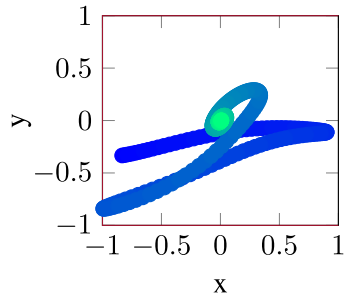

Fig. 9. The path of the bearing to the target in the image, starting on the left and ending up centered. The image boundary is given by the solid boundaries. We observe that the field of view constraints are not violated.

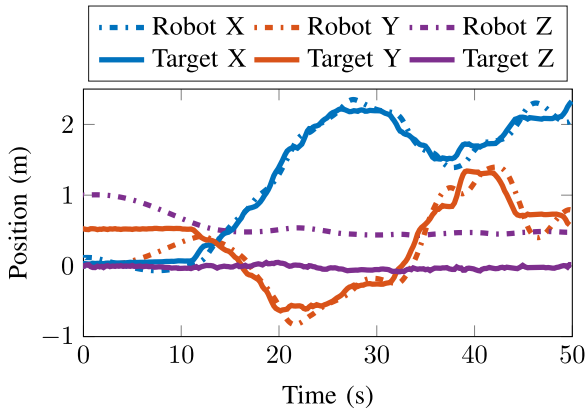

Fig. 10. Experimental results of a robot tracking a rolling sphere. The vehicle quickly locks onto the target and maintains the desired relative pose despite erratic motion in the target's path.

constant-velocity target with an initial state of

$$\mathbf{x} = \begin{bmatrix} -2.5 \\ 1 \\ 0 \end{bmatrix}, \quad \dot{\mathbf{x}} = \begin{bmatrix} 5 \\ -1 \\ 0 \end{bmatrix}$$

and the robot's initial state given by

$$\mathbf{x} = \begin{bmatrix} 0 \\ 0 \\ 3 \end{bmatrix}, \quad \dot{\mathbf{x}} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \quad R = \mathbf{I}, \quad \boldsymbol{\Omega} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}.$$

The planning horizion is 1 second with a trajectory update frequency of 10 Hz. The desired relative pose is defined such that the robot is 1 m above the target, and the horizontal and vertical fields of view are both assumed to be 90°. The resultant trajectory is plotted in Fig. 7. Each new planned trajectory's velocity is plotted in Fig. 8. The resultant path of the object in the image is plotted in Fig. 9. We see some very exciting results. For example, despite the fact that there is no initial velocity error in the z direction, the robot accelerates upward, helping to keep the object in the field of view (Fig. 8). With the same initial conditions and minimizing the position error from the start, the motion is quickly dominated by the visual constraints and results in an infeasible problem.

Next we present our experimental results, which were executed in the GRASP (General Robotics Automation Sensing and Perception) lab at the University of Pennsylvania. The total flying area has a volume of $20 \times 6 \times 4\,\mathrm{m}^3$. We implemented the entire estimation, planning, and control pipeline, including the robot state estimator, the target's state estimator, a target trajectory predictor, and the trajectory planning algorithm on a Qualcomm Snapdragon board, featuring a Qualcomm Hexagon DSP and 802.11n Wi-Fi, all packed onto a (58 × 40 mm) board based on the Snapdragon 801 processor. A Kalman filter is used for the robot's state estimation at 500 Hz with respect to a fixed reference frame. The estimation is obtained combining IMU data in the prediction phase and distinguishable environment landmarks in the measurement update step. For more details, we refer the reader to our previous work, where reliable flights with speeds up to 5 m/s and angular rates of 800 deg/s are achieved [36]. The quadrotor is only 250 g, and a single downward-facing camera and onboard IMU are used to perform both the state estimation of the robot and the target.

The target is a 7.6 cm diameter Sphero SPRK+ spherical robot, controllable using a smartphone via Bluetooth. Our estimator fits a polynomial trajectory to the 10 latest target pose measurements (at 30 Hz). For these experiments, the estimator uses a constant model in the z direction since the target is rolling on a flat surface. The robot replans frequently enough that a first order regression model in the horizontal directions is sufficient to estimate the motion. Then, the estimated trajectory is sampled over the next planning horizon to determine the coefficients $\mathbf{h}$ for the next segment.

With this platform, we demonstrate successful results such as the tracking in Fig. 10 and the acceleration to track a quickly moving target entering the field of view in Fig. 1. In both cases, the object does not leave the field of view, and the quadrotor successfully tracks the target. For more results, including simulations of the approach applied to objects moving in 3D, we refer the reader to the attached video or to the higher-resolution video at http://www.jtwebs.net/2017-ra-l/.

VI. Conclusion

This work presented a relative pose estimation and trajectory planning strategy to track a moving sphere with an underactuated micro aerial vehicle while considering the dynamic, actuator, and field of view constraints. We validated that all perception and computation can occur onboard a 250 g robot equipped with only one downward-facing camera and an inertial measurement unit. Simulation and experimental results demonstrate successful tracking of a moving target while keeping the object in the field of view.

There are many exciting research opportunities stemming from this work. For example, an online scale or range estimate is needed, which could be inspired from the strategies used by dragonflies. Optical flow (considering the parallax) between the target and background could also help improve the estimate of the relative velocities. Finally, a better-informed strategy is needed to select the cost function weightings, λ_i.

References

[1] T. Tomic *et al.*, "Toward a fully autonomous UAV: Research platform for indoor and outdoor urban search and rescue," *IEEE Robot. Autom. Mag.*, vol. 19, no. 3, pp. 46–56, Sep. 2012.

[2] T. Ozaslan, S. Shen, Y. Mulgaonkar, N. Michael, and V. Kumar, "Inspection of penstocks and featureless tunnel-like environments using micro UAVs," in *Proc. Field Serv. Robot. Conf.*, Brisbane, Qld, Australia, 2013, pp. 123–136.

[3] J. Cacace, A. Finzi, V. Lippiello, G. Loianno, and D. Sanzone, *Aerial Service Vehicles for Industrial Inspection: Task Decomposition and Plan Execution*. Berlin, Germany: Springer, 2013, pp. 302–311.

[4] G. Loianno, J. Thomas, and V. Kumar, "Cooperative localization and mapping of MAVS using RGB-D sensors," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2015, pp. 4021–4028.

[5] J. Thomas, G. Loianno, K. Daniilidis, and V. Kumar, "Visual servoing of quadrotors for perching by hanging from cylindrical objects," *IEEE Robot. Autom. Lett.*, vol. 1, no. 1, pp. 57–64, Jan. 2016.

[6] N. Michael *et al.*, "Collaborative mapping of an earthquake-damaged building via ground and aerial robots," *J. Field Robot.*, vol. 29, no. 5, pp. 832–841, 2012.

[7] J. Kim, Y. Jung, D. Lee, and D. H. Shim, "Outdoor autonomous landing on a moving platform for quadrotors using an omnidirectional camera," in *Proc. Int. Conf. Unmanned Aircr. Syst.*, Orlando, FL, USA, May 2014, pp. 1243–1252.

[8] J. Thomas, J. Polin, K. Sreenath, and V. Kumar, "Avian-inspired grasping for quadrotor micro UAVs," in *Proc. Int. Des. Eng. Tech. Conf. Comput. Inf. Eng. Conf.*, Portland, OR, USA, Aug. 2013, Paper DETC2013-13289.

[9] J. Thomas *et al.*, "Planning and control of aggressive maneuvers for perching on inclined and vertical surfaces," in *Proc. Int. Des. Eng. Tech. Conf. Comput. Inf. Eng. Conf.*, Boston, MA, USA, 2015, pp. 1–10.

[10] R. T. Fomena and F. Chaumette, "Visual servoing from spheres using a spherical projection model," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 2007, pp. 2080–2085.

[11] R. Fomena and F. Chaumette, "Improvements on visual servoing from spherical targets using a spherical projection model," *IEEE Trans. Robot.*, vol. 25, no. 4, pp. 874–886, Aug. 2009.

[12] D. Eberli, D. Scaramuzza, S. Weiss, and R. Siegwart, "Vision based position control for MAVs using one single circular landmark," *J. Intell. Robot. Syst.*, vol. 61, no. 1–4, pp. 495–512, Jan. 2011.

[13] F. P. Adler, "Missile guidance by three-dimensional proportional navigation," *J. Appl. Phys.*, vol. 27, no. 5, pp. 500–507, 1956.

[14] S. A. Murtaugh and H. E. Criel, "Fundamentals of proportional navigation," *IEEE Spectr.*, vol. 3, no. 12, pp. 75–85, Dec. 1966.

[15] K. E. Wenzel, A. Masselli, and A. Zell, "Automatic take off, tracking and landing of a miniature UAV on a moving carrier vehicle," *J. Intell. Robot. Syst., Theory Appl.*, vol. 61, no. 1–4, pp. 221–238, 2011.

[16] D. Lee, T. Ryan, and H. J. Kim, "Autonomous landing of a VTOL UAV on a moving platform using image-based visual servoing," in *Proc. IEEE Int. Conf. Robot. Autom.*, May. 2012, pp. 971–976.

[17] B. Herisse, T. Hamel, R. Mahony, and F.-X. Russotto, "Landing a VTOL unmanned aerial vehicle on a moving platform using optical flow," *IEEE Trans. Robot.*, vol. 28, no. 1, pp. 77–89, Feb. 2012.

[18] J. Chen, T. Liu, and S. Shen, "Tracking a moving target in cluttered environments using a quadrotor," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2016, pp. 446–453.

[19] W. Gander, G. H. Golub, and R. Strebel, "Least-squares fitting of circles and ellipses," *BIT Numer Math.*, vol. 34, no. 4, pp. 558–578, Dec. 1994.

[20] A. Fitzgibbon and R. Fisher, "A buyer's guide to conic fitting," in *Proc. Br. Mach. Vis. Conf.*, 1995, pp. 51.1–51.10.

[21] O. Faugeras, *Three-dimensional Computer Vision: A Geometric Viewpoint*. Cambridge, MA, USA: MIT Press, 1993.

[22] E. Marchand, F. Spindler, and F. Chaumette, "ViSP for visual servoing: A generic software platform with a wide class of robot control skills," *IEEE Robot. Autom. Mag.*, vol. 12, no. 4, pp. 40–52, Dec. 2005.

[23] R. Spica, P. R. Giordano, and F. Chaumette, "Active structure from motion for spherical and cylindrical targets," in *Proc. IEEE Int. Conf. Robot. Autom.*, Hong Kong, May 2014, pp. 5434–5440.

[24] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2011, pp. 2520–2525.

[25] T. Lee, M. Leoky, and N. H. McClamroch, "Geometric tracking control of a quadrotor UAV on SE(3)," in *Proc. IEEE Conf. Decis. Control.*, Dec. 2010, pp. 5420–5425.

[26] J. Thomas *et al.*, "Aggressive flight for perching on inclined surfaces," *J. Mech. Robot.*, vol. 8, no. 5, May. 2016, Art. no. 051007.

[27] R. M. Olberg, A. H. Worthington, and K. R. Venator, "Prey pursuit and interception in dragonflies," *J. Comparative Physiol., Sensory, Neural, Behav. Physiol.*, vol. 186, no. 2, pp. 155–162, Feb. 2000.

[28] S. A. Kane and M. Zamani, "Falcons pursue prey using visual motion cues: New perspectives from animal-borne cameras," *J. Exp. Biol.*, vol. 217, no. 2, pp. 225–234, Jan 2014.

[29] M. McBeath, D. Shaffer, and M. Kaiser, "How baseball outfielders determine where to run to catch fly balls," *Science*, vol. 268, no. 5210, pp. 569–573, Apr. 1995.

[30] L. Peper, R. J. Bootsma, D. R. Mestre, and F. C. Bakker, "Catching balls: How to get the hand to the right place at the right time," *J. Exp. Psychol. Human Perception Perform.*, vol. 20, no. 3, pp. 591–612, Jun. 1994.

[31] M. Turpin, N. Michael, and V. Kumar, "Trajectory design and control for aggressive formation flight with quadrotors," *Auton. Robots*, vol. 33, no. 1–2, pp. 143–156, Aug. 2012.

[32] B. Siciliano and O. Khatib, Eds., *Springer Handbook of Robotics*. Berlin, Germany: Springer, 2008.

[33] M. Mueller and R. D'Andrea, "A model predictive controller for quadrocopter state interception," in *Proc. Eur. Control Conf.*, 2013, pp. 1383–1389.

[34] S. G. Johnson, "The NLopt nonlinear-optimization package." 2017. [Online]. Available: http://ab-initio.mit.edu/nlopt

[35] P. E. Gill and E. Wong, *Mixed Integer Nonlinear Programming* (The IMA Volumes in Mathematics and its Applications Series), vol. 154, J. Lee and S. Leyffer, Eds. New York, NY, USA: Springer, 2012.

[36] G. Loianno, C. Brunner, G. McGrath, and V. Kumar, "Estimation, control, and planning for aggressive flight with a small quadrotor with a single camera and IMU," *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 404–411, Apr. 2017.