Distributed vision-based flying cameras to film a moving target

Fabio Poiesi and Andrea Cavallaro

***Abstract*— Formations of camera-equipped quadrotors (flying cameras) have the actuation agility to track moving targets from multiple viewing angles. In this paper we propose an infrastructure-free distributed control method for multiple flying cameras tracking a moving object. The proposed vision-based servoing can deal with noisy and missing target observations, accounts for quadrotor oscillations and does not require an external positioning system. The flight direction of each camera is inferred via geometric derivation, and the formation is maintained by employing a distributed algorithm that uses the target position information on the camera plane and the position of neighboring flying cameras. Simulations show that the proposed solution enables the tracking of a moving target by the cameras flying in formation despite noisy target detections and when the target is outside some of the fields of view.**

I. Introduction

The miniaturization of smart cameras with improved computational capabilities at lower power consumption enables sophisticated on-board processing [1]. A smart camera can be mounted on a small quadrotor (we refer to the whole as a *flying camera*) to support its autonomous control either as a stand-alone device or in a team. Unlike fixed wing robotic platforms that need to continuously fly around a certain location, quadrotors are more agile and can hover thus facilitating video-based operations.

Most methods for the control of teams of quadrotors rely on external positioning and control systems based on motion capture [2] or GPS [3]. Motion capture or GPS provide reliable positioning information, but do not offer the flexibility to use flying cameras in diverse scenarios. To address this problem, Shen et al. [4] proposed an heterogeneous sensor fusion approach for both indoor and outdoor navigation. Preliminary experiments for autonomous systems are still based on constraining assumptions, such as the use of markers or manual intervention [5], [6].

In this paper, we propose a method for tracking a moving target with formations of cameras flying without an external control system. The method relies only on the detected target on the camera plane and on the relative positions of neighboring flying cameras. The video processing used for flight control has to tackle problems of noisy measurements of the target position, which lead to noisy flight controls, and oscillations of the quadrotor itself, which in turn lead to noisy measurements of the target, or its loss. Unlike [7] and [8], no measurements of the relative 3D distance or position between the target and the flying cameras' body coordinate system are needed. From an application view point relative positions can be obtained using a 3D proximity sensor [9] or ultrasound [10]. We propose a thrust control method running on each flying camera that adapts its velocity according to the velocity inferred from the moving target on the camera plane. In order to be robust against noisy target detections and to flying camera oscillations, the flight velocities are set using the knowledge exchanged with neighbors in a distributed manner. The formation shape is maintained over time by geometrically constraining it via the circle intersection rule [11]. The method is evaluated via simulations using formations up to 12 flying cameras, where the moving target follows arbitrary trajectories. We show that the flying cameras can track the target by maintaining the formation and autonomously recover when the target goes out of some fields of view.

*This work was supported by the Artemis JU and the UK Technology Strategy Board through COPCAMS Project under Grant 332913.

Fabio Poiesi and Andrea Cavallaro are with Centre for Intelligent Sensing, Queen Mary University of London, Mile End Road, E1 4NS, London, United Kingdom. {fabio.poiesi,a.cavallaro}@qmul.ac.uk

II. Related work

Quadrotors can perform cooperative surveillance [12], [13], best camera coverage [14], [15], SLAM [16], target detection and tracking [17], [18], 3D target enclosing [7] and mosaicing [19].

In a *cooperative surveillance* task, flying cameras can maximize the view of a set of areas of interest based on the optimization of an objective function reflecting the quality of the surveillance task [13]. The optimization can be calculated with Particle Swarm Optimization [20] under motion constraints with respect to absolute and relative flying camera positions. Cooperative surveillance also tackles the multi-target search problem [12]. A probability map is built to provide information about likely locations of detected targets and is temporally updated by each flying camera. The measurements with which the probability map is constructed are obtained via consensus algorithm where the local measurements are shared among neighbors. Area coverage and detection performance (i.e. false alarms) are taken into account in the optimization process.

When the task is *best camera coverage* the amount of visible parts of the environment have to be maximized, and the distance between a flying camera and a point in the environment has to be minimized [15]. A team of flying cameras can be positioned so that the union of their field of views maximizes the visual coverage of a planar environment [14]. This can be achieved by minimizing a cost function expressing the information per pixel within the field of view. The flight controller is robust to addition or to deletion from the team of flying cameras. Alternatively, best camera coverage can be achieved with the 3D map of the environment obtained via SLAM [15].

Target detection and tracking can be performed from a single flying (infrared) camera by background subtraction and gradient-based features for detection, and particle filter for tracking [17]. Multiple targets can also be tracked with multiple flying cameras [18]. Because a single flying camera at a high altitude can track multiple targets, but with a lower view quality than a lower altitude camera, Tokekar et al. [18] find a trade-off between the number of tracked targets assigned to a camera and the corresponding tracking quality.

3D target enclosing aims at positioning flying cameras around a target [7]. The enclosing method uses the relative positions between the flying cameras and the target to move the cameras to their final positions. This method relies only on the body coordinate system of each camera, but the final positions around a target need to be defined at initialization. The movements are achieved by rotating and translating the flying cameras in order to minimize the quadratic error between starting and final positions. The Kabsch algorithm is used to perform this minimization [21]. Our method differs from [7] as we do not set any final position around the target and we let the flying cameras maintain the formation on target via their vision-based control. Also, we use real flight dynamics of quadrotors to follow the target, whereas in [7] the dynamics are not taken into account.

III. Problem Formulation and Definitions

Let us consider a formation of N flying cameras (i.e. smart cameras mounted on quadrotors) capable of detecting a moving object (target) and having only the information of the target position on the 2D camera plane available.

The target trajectory point is defined as $x_t(k) \in \mathbb{R}^3$, which is the position of the target in the global coordinate system at k. The subscript t stands for *target*.

The camera plane is of size $[-\frac{W}{2}\ \frac{W}{2}]\times[-\frac{H}{2}\ \frac{H}{2}]$, where W and H are the width and height, respectively. Each camera has angle of view ϕ and focal length f_L.

Let us define the flying camera formation as the graph $F(k) = (\mathcal{C}(k), D(k))$. $\mathcal{C}(k) = \{C_i(k)\}_{i=1}^{N}$ at time k, where $C_i(k)$ is the state of the i^{th} flying camera at $k \in \mathbb{R}$. $D(k) \in \mathbb{R}^{N\times N}$ is the graph that defines the formation in terms of connectivity and distances among $\mathcal{C}(k)$. D_i is the set of neighbors of $C_i(k)$, and $d_{i,j}(k) \in D(k)$ defines the distance between $C_i(k)$ and $C_j(k)$. The state $C_i(k) = (x_i(k), R_i(k))$ defines the position $x_i(k) \in \mathbb{R}^3$ in the global coordinate system and attitude $R_i(k) \in SO(3)$ [22]. $R_i(k)$ is the rotation matrix from the body (local) to the global coordinate system that defines the flying camera's attitude. The body is defined by three main body directions $b_{1,i}(k) = R_i(k)e_1$, $b_{2,i}(k) = R_i(k)e_2$ and $b_{3,i}(k) = -R_i(k)e_3$, where $e_1 = (1,0,0)^T$, $e_2 = (0,1,0)^T$ and $e_3 = (0,0,1)^T$. Each flying camera has velocity $v_i(k) = (v_{x,i}(k), v_{y,i}(k), v_{z,i}(k))$ in the global coordinate system. At $k = 0\ \forall i$, $C_i(0)$ and $v_i(0)$ are given, and each camera points towards the target with a fixed orientation $R_{c,i}$ with respect to the body coordinate system $R_i(k)$. The target is detected by each flying camera and its position on the i^{th} camera plane is $\tilde{x}_{t,i}(k) \in \mathbb{R}^2$. The target velocity is defined as $v_t(k)$. Let us assume that the target is moving on a plane with $||v_t(0)|| \sim ||v_i(0)||$ and with a direction within the range of $180°$ of the flying cameras' heading direction. The flying camera's motion is determined by a controller that follows a goal (desired) trajectory point $x_{g,i}(k) \in \mathbb{R}^3$, and a goal (desired) direction of the first body direction $b_{1_g,i}(k)$ [22]. The subscript g stands for *goal*.

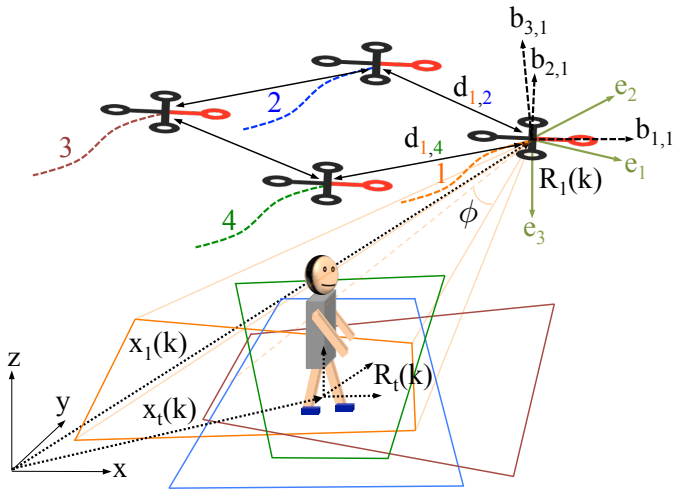

Fig. 1. Formation of four flying cameras filming a moving target. Target in position $x_t(k)$ with orientation $R_t(k)$. Fields of view of each camera are represented on the ground plane with different colors. Flying camera 1 shows an example of the variables that characterize its attitude and formation constraints (distances from neighbors).

We assume that the flying cameras mount a proximity sensor capable of estimating the position of neighboring flying cameras (e.g. [9], [10]). The cameras are initialized on the target and we aim to track the target while maintaining the flying cameras in formation. Similarly to [6], flying cameras operate on a horizontal plane at the same altitude. Measurements of the altitude can be carried out with an onboard barometer [23]. We assume scenarios with absence of atmospheric perturbations such as wind. If the target is lost by all the cameras there is no strategy to search for it. Fig. 1 summarizes the variables.

IV. Vision-based Thrust Control

The goal is to regulate the velocity of a flying camera by exploiting the target position on the camera plane. Intuitively, if we are tracking a target from a flying camera, we can exploit the location of the target on the camera plane to infer whether the flying camera's speed differs from that of the target. For example, with a camera pointing downwards, if the target is positioned on the lower part of the camera plane, it means that the flying camera needs to slow down. If the target is positioned on the right-hand side of the camera, the flying camera should turn right to maintain the view of the target. More generally, the goal is to adapt the flying-camera dynamics to keep the target at the center of its field of view. Therefore, we design a thrust control method that allows a formation of flying cameras to autonomously track a moving target. We aim to trigger the thrust control method

only when $\tilde{x}_{t,i}(k)$ moves away from the center of the image plane.

The thrust control is divided into two steps, namely computations of the gain and the direction. The *thrust gain* indicates the velocity amplification to be applied in a certain direction. The thrust gain is computed via a function $M : \mathbb{R}^2 \rightarrow \mathbb{R}$ that maps the target position on the camera plane to a scalar value $m_i(k)$ expressing the amplification of $C_i(k)$. M is designed such that the thrust gain increases as the distance from the center of the camera plane increases and vice versa. The trend of M is regulated via the covariance matrix $\Sigma_m \in \mathbb{R}^{2\times 2}$ as

$$m_i(k) = M(\tilde{x}_{t,i}(k)) = 1 - \exp\left(-\frac{1}{2}\tilde{x}_{t,i}(k)^T \Sigma_m^{-1} \tilde{x}_{t,i}(k)\right). \tag{1}$$

The farther the target from the center, the larger $m_i(k)$. For example, if the target is in the center of the field of view $\tilde{x}_{t,i}(k) = [0\ 0]^T \rightarrow m_i(k) = 0$, whereas if the target is on the right-hand side border $\tilde{x}_{t,i}(k) = [\frac{W}{2}\ 0]^T \rightarrow m_i(k) = 1 - \exp\left(-\frac{1}{2}[\frac{W}{2}\ 0]^T \Sigma_m^{-1} [\frac{W}{2}\ 0]\right)$.

The *thrust direction* is computed by projecting $\tilde{x}_{t,i}(k)$ onto two virtual axes on the camera plane that are used as a reference to infer the target dynamics. The origin of the virtual axes is located in the center of the camera plane, and are defined as $e_{x,i} = R_{c,i}e_1$ and $e_{y,i} = R_{c,i}e_2$, where $R_{c,i}$ is the fixed orientation of the camera with respect to $R_i(k)$ (Sec. III). The thrust direction is thus computed as

$$\begin{aligned} a_{x,i}(k) &= sgn(\tilde{x}_{t,i}(k) \cdot e_{x,i}) m_i(k) \\ a_{y,i}(k) &= sgn(\tilde{x}_{t,i}(k) \cdot e_{y,i}) m_i(k), \end{aligned} \tag{2}$$

where $sgn()$ is the *sign* function and $\cdot$ is the dot product.

When $\tilde{x}_{t,i}(k)$ moves towards the center of the camera plane, the vision-based thrust control must stop its action as it is enough to get $\tilde{x}_{t,i}(k)$ centered in the field of view of the flying camera. To decide when to stop this thrust action, we compute the difference over time

$$\begin{aligned} \dot{a}_{x,i}(k) &= |a_{x,i}(k)| - |a_x(k - \tau_a \Delta_k)| \\ \dot{a}_{y,i}(k) &= |a_{y,i}(k)| - |a_y(k - \tau_a \Delta_k)|, \end{aligned} \tag{3}$$

where $|\ |$ indicates the absolute value, τ_a regulates the time interval when to compute the difference and Δ_k is the sampling time. The larger τ_a, the more robust the thrust control to target speed variations or noisy detections. The smaller τ_a, the quicker the reaction of the flying camera to target speed variations. Note that, a sensitive reaction to speed variations may cause oscillations due to sudden accelerations of the flying cameras, with the risk of losing the target from their fields of view. A smooth reaction may not be sufficient to quickly adapt to the target speed and the target may be lost. The reaction of the flying cameras at changing values of τ_a is evaluated in Sec. VI.

The thrust commands inferred on the camera plane are translated into body control commands by the flying camera. As the cameras are flying at the same altitude, the commands are about the first and second body directions, $b_{1,i}(k)$ and $b_{2,i}(k)$. Eq. 1 and 3 predict each flying-camera's velocity as a function of the measured target's dynamics on the camera plane. For each $C_i(k)$ the predicted velocity is

$$\begin{cases} v'_{x,i}(k) = \alpha a_{x,i}(k) + v_{x,i}(k - \Delta_k) & \text{if } \dot{a}_{x,i}(k) > 0 \\ v'_{x,i}(k) = v_{x,i}(k - \Delta_k) & \text{otherwise} \\ \\ v'_{y,i}(k) = \alpha a_{y,i}(k) + v_{y,i}(k - \Delta_k) & \text{if } \dot{a}_{y,i}(k) > 0 \\ v'_{y,i}(k) = v_{y,i}(k - \Delta_k) & \text{otherwise,} \end{cases} \tag{4}$$

where α amplifies the thrust gain and defines the desired behavior of the flying cameras to target dynamics variations. The larger α, the higher the sensitivity to target dynamics variations. We perform the sensitivity analysis of α in Sec. VI.

In the next section we explain how each $C_i(k)$ updates its own velocity in a distributed manner by using the information retrieved from their neighbors.

V. Distributed formation control

In a flying formation there must exist some sort of agreement on the velocity to maintain. The velocity prediction of some flying cameras in the formation can be challenging to be computed accurately due to noisy target detections and/or oscillations of the quadrotor. A distributed algorithm with consensus on the whole formation cannot be used as flying cameras should be able to slightly adapt/change their velocity in order to allow maneuvers. An example is when the formation has to keep track of a target turning left. Flying cameras on the right-hand side of the formation must set a larger velocity to accomplish the maneuver than those on the left-hand side. Moreover, the velocity update must also guarantee that the formation geometry is maintained over time, that is the distances between each pair of flying cameras must remain the same.

To achieve this, each $C_i(k)\ \forall k$ communicates to its neighbors $v'_i(k)$ and updates $v_i(k)$ using the velocity received from the neighbors D_i as

$$\begin{aligned} v_{x,i}(k) &= \frac{1}{card(D_i)} \sum_{j \in D_i} v'_{x,j}(k) \\ v_{y,i}(k) &= \frac{1}{card(D_i)} \sum_{j \in D_i} v'_{y,j}(k), \end{aligned} \tag{5}$$

where $card()$ is the cardinality of a set, D_i is the set of neighbors and $v_i(k) = (v_{x,i}(k), v_{z,i}(k), v_{z,i}(k))^T$ (Sec. III). We consider $v_{z,i}(k) = v_{z,i}(0)$ that remains unchanged over time. Note that in practice the value of the velocity can have fluctuations due to errors in the measurement of the altitude [6], [24]. These variations could be taken into account by looking at the region extent of the detected target on the image plane.

The velocity $v_i(k)$ provides only an indication of each individual $C_i(k)$'s desired trajectory (i.e. heading direction). We might need to adjust the velocity in order to follow the constraints of the formation shape. We use a non-hierarchical formation control based on the circle intersection rule to impose geometric constraints. The circle intersection rule

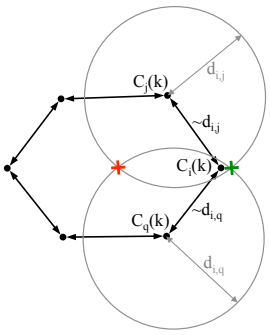

Fig. 2. Formation topology where the distances between flying cameras may be slightly different than those defined at initialization. The circles intersection rule allows flying cameras to be constrained to the formation shape. In this example the green intersection is selected as it is the closest to $C_i(k)$.

requires at least two neighbors per flying cameras in order to obtain the intersection points (Fig. 2). Unlike leader-follower control, non-hierarchical control is more robust to external noise (e.g. noisy target detections) and to speed variations of individual flying cameras [11]. Also, non-hierarchical control allows an even distribution of target tracking and formation preservation among the flying cameras.

Given the relative position of neighbors with respect to the body reference and updated velocity $v_i(k)$, $C_i(k)$ computes the desired trajectory $x_{g,i}(k)$ to maintain itself in formation. The algorithm first computes a candidate desired trajectory, and then confirms or adjusts it based on the circle intersection rule [11]. The candidate desired trajectory is calculated as

$$x^c_{g,i}(k) = x_i(k - \Delta_k) + v_i(k)\Delta_k. \tag{6}$$

To confirm $x^c_{g,i}(k)$ as the desired trajectory that follows the constraints of the formation shape, we compute the point

$$\begin{aligned} p^c_i(k) = \arg\min_x \{||x - x^c_{g,i}(k)||\} \;\; s.t. \\ x \in \Gamma(x^c_{g,j}(k), d_{i,j}(0)) \cap \Gamma(x^c_{g,q}(k), d_{i,q}(0)), \\ j, q \in D_i, j \neq q, \end{aligned} \tag{7}$$

where $\Gamma(x, d)$ defines the circle with center x and radius d, $d_{i,j}(0)$ is the distance between $C_i(k)$ and $C_j(k)$ set at $k = 0$ (Sec. III). $x^c_{g,j}(k)$ and $x^c_{g,q}(k)$ are computed by $C_i(k)$ using the measured positions at $k-1$ and the updated velocities received from $C_j(k)$ and $C_q(k)$ at k, respectively. $\Gamma(x^c_{g,j}(k), d_{i,j}) \cap \Gamma(x^c_{g,q}(k), d_{i,q})$ is assumed to be non-empty. Please refer to [11] for details about the empty intersection case. The desired trajectory is then computed as

$$x_{g,i}(k) = \begin{cases} x^c_{g,i}(k) & \text{if } \epsilon_{i,j} > \varepsilon \text{ or } \epsilon_{i,q} > \varepsilon \\ p^c_i(k) & \text{otherwise,} \end{cases} \tag{8}$$

with

$$\begin{aligned} \epsilon_{i,j} &= ||x^c_{g,i}(k) - x^c_{g,j}(k)|| - d_{i,j}(0) \\ \epsilon_{i,q} &= ||x^c_{g,i}(k) - x^c_{g,q}(k)|| - d_{i,q}(0), \end{aligned} \tag{9}$$

where $||\ ||$ is the ℓ-2 norm and ε is the separation tolerance term. The larger ε, the more constrained the flying cameras in maintaining the inter-distances defined at $k = 0$, $D(0)$.

The desired first-body direction points towards the desired trajectory and it is computed as

$$b_{1_{g,i}}(k) = x_{g,i}(k) - x_i(k - \Delta_k). \tag{10}$$

The dynamics of the camera-equipped quadrotors are determined by four identical propellers, which are equidistant from the centre of body, and generate a thrust and torque normal to the body plane defined by $b_{1,i}(k)$ and $b_{2,i}(k)$ [22]. Given a desired trajectory point $x_{g,i}(k)$, the total thrust applied to a flying camera and the desired direction of the third-body axis $b_{3_{g,i}}(k)$ are selected to stabilize the translational dynamics. Please refer to [22], [8] for a detailed explanation of the model for the control method.

VI. Simulation Results

In this section we discuss the behavior of the flying cameras when tracking a moving target with the proposed method. We quantify the performance of the proposed method in terms of ratios of cameras viewing the target (i.e. the average and minimum number of cameras having the target included in their camera plane). We also analyze the sensitivity of the proposed approach to variations of κ, τ_a, initial velocity $v_i(0)$, and to noise and miss target detections. The noise applied to the target position is additive Gaussian noise with standard deviation $\sigma_n = p_n W$, where $p_n \in [0\ 0.1]$, and $p_m \in [0\ 0.5]$ defines the probability of missing a target per frame. Lastly, we show qualitative results by filming the target from the view of each flying camera and by automatically generating a video using the best view over time.

Let each quadrotor mount a camera with focal length $f_L = 0.2$ and angle of view $\phi = \frac{\pi}{5}$. The resolution of the camera is $W = H = 100$. We evaluate the proposed method with formations of $N = 6, 8, 10, 12$ flying cameras. Our goal is to maintain the center of the formation along the *x*-*y* plane on the target position, which correspond to the target centered on each camera plane. The starting positions of the cameras are

$$x_i(0) = \left(L\, cos((i-1)\frac{\pi}{N/2}),\ L\, sin((i-1)\frac{\pi}{N/2}),\ A \right)^T, \tag{11}$$

where $L = 15$ is the radius of the formation and $A = 10$ is the altitude. The initial velocity is $v_i(0) = (2\Delta_k, 0, 0)^T\ \forall i$, with $\Delta_k = 0.04$ for all the experiments. Each camera is pointed towards the target with respect to its respective initial orientation $R_i(0)$. The graph $D(k)$ is defined such that each $C_i(k)$ is only aware of its closest neighbors (e.g. Fig. 2) and $d_{i,j}(k)$ is to be maintained over time according Eq. 11. We use the following values: $\tau_a = 4$ (Eq. 3), $\varepsilon = 0.1$ (Eq. 8) and $\alpha = 0.05$ (Eq. 4), and we will also discuss results with different values of α and τ_a. $\Sigma_m = diag(50, 50)$ (Eq. 1). The parameters of the quadrotor's dynamics are set as in [8].

We test the control method on four different trajectories: linear (LT): $v_t(k) = (3\Delta_k, 0, 0)^T$, sinusoidal (ST): $v_t(k) = (2\Delta_k, 0.08cos(\frac{2}{5}\pi k), 0)^T$, and two manually drawn (T1 and T2) with respective average velocities (T1) $v_t(k) = (2.5\Delta_k, -0.24\Delta_k, 0)^T$ and (T2) $v_t(k) =$

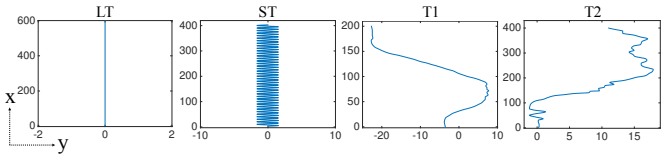

Fig. 3. Target trajectories used for the simulations. LT: linear (8s - 200 time steps); ST: sinusoidal (8s - 200 time steps); T1: trajectory 1 (80s - 2000 time steps); T2: trajectory 2 (186.64s - 4666 time steps).

$(2.15\Delta_k, 0.06\Delta_k, 0)^T$. The duration of LT and ST is 8s (200 time steps), T1 is 80s (2000 time steps) and T2 is 186.64s (4666 time steps). Fig. 3 shows the trajectories.

The experiments on LT and ST provide perfect results in terms of ratio of flying cameras viewing the target. This improves the performance obtained in [8] where the flying cameras in ST could not constantly view the target. This is achieved thanks to the visual feedback that the proposed approach uses for the flying cameras' positioning. The smoothing function M allows each camera to react to the target motion only when important maneuvers are required, e.g. turns, and to maintain the main trajectory in situations when a target follows a certain direction but with mild orientation variations (e.g. oscillations). A different attitude occurs in Fig. 4 that shows the ratio of the flying cameras viewing the target over time for T1 and T2. The cameras view the target also when it undergoes large motion variations. The average ratio (see legend Fig. 4a) shows that the proportion of the cameras viewing the target is roughly constant. Before time step 1000 (the curve in T1 before time step 100) all the formations cannot maintain their views on target with all the cameras. This happens because of two reasons. First, the dynamic of the target is close to the limit of the assumptions of our method (range of $180°$ with respect the heading direction). Second, the vision-based control needs to provide a vigorous response to the cameras to perform the first turn in order to quickly position each camera on target before they accomplish the second turn. A vigorous response can be achieved with a large value of α (Eq. 4). However, having a large α would lead to a higher positioning sensitivity that may cause formation instabilities in the case of noise along the trajectory, e.g. T2 in Fig. 3. Moreover, it is interesting to observe that when the target goes out of the majority of the fields of view (60% - case $N = 10$ Fig. 4a), the formation can recover and reconfigure itself on target with all the flying cameras fully viewing the moving target.

We assess the sensitivity of the vision-based control to variations of α and τ_a values. In the case of α we can observe from Fig. 5 that large values of α lead to situations where the average ratio of flying cameras viewing a target differs for different formations (T1) and decreases (T2). Performance drops more quickly in T2 because it is noisier than T1 (Fig. 3). There are cases where for large values of α the performance improves, but we chose the value of α providing stable performance across different formations. In the case of τ_a the formation behaviors are stable across different values of τ_a (Fig. 6). The lower the value of τ_a (i.e. 1 or 2), the

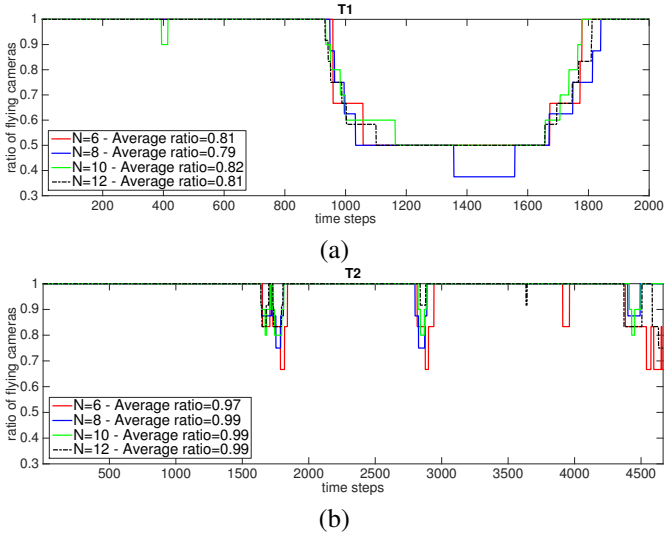

Fig. 4. Ratio of flying camera filming a moving target along different trajectories: (a) T1 and (b) T2. Results show that the vision-based control is more robust to high-frequency motion variations than those at low-frequency.

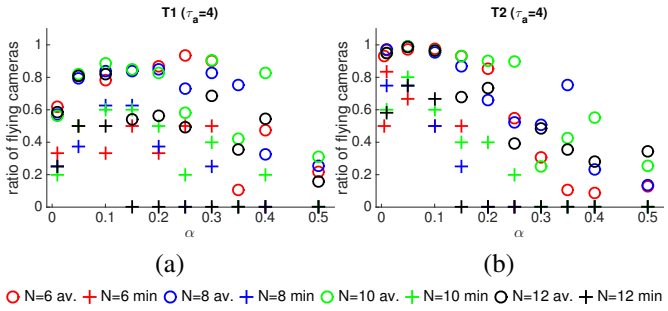

Fig. 5. Sensitivity analysis for variations of α. The vertical axis represents the average (circle marker) and minimum (cross marker) ratios of cameras viewing the moving target over time. The horizontal axis represents the variations of α.

worse on average the view of the target. Large values of τ_a are then more suitable in the case of noisy target dynamics changes.

Fig. 7 shows the average ratio of flying cameras viewing the moving target over time when Gaussian noise is added to the target position on each camera plane independently. The vision-based control is robust to noise in the case of T2. In T1 our method fails to maintain the cameras on target when they turn as the noise makes them drift. Across the graphs we can observe cases where the minimum number of flying cameras viewing the target goes below 0.2 (situations with only one flying camera viewing the target) but with a high average ratio, which means that the formation could track the target throughout the trajectory.

Fig. 8 shows a sample experiment (from an application point of view) on T2 at time instant 3500, where the flying cameras and target's trajectories are drawn. From Fig. 4 we can relate two instances (after time instant 1500 and before 3000) where some flying cameras lost the view of the target in Fig. 8. These two cases correspond to the two curves along the target's trajectory, which are also reflected on the flying

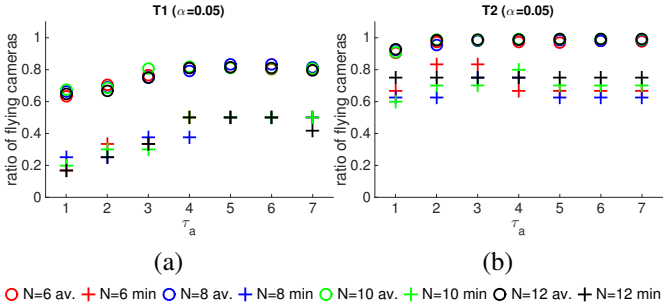

Fig. 6. Sensitivity analysis for variations of τ_a. The vertical axis represents the average (circle marker) and minimum (cross marker) ratios of cameras viewing the moving target over time. The horizontal axis represents the variations of τ_a.

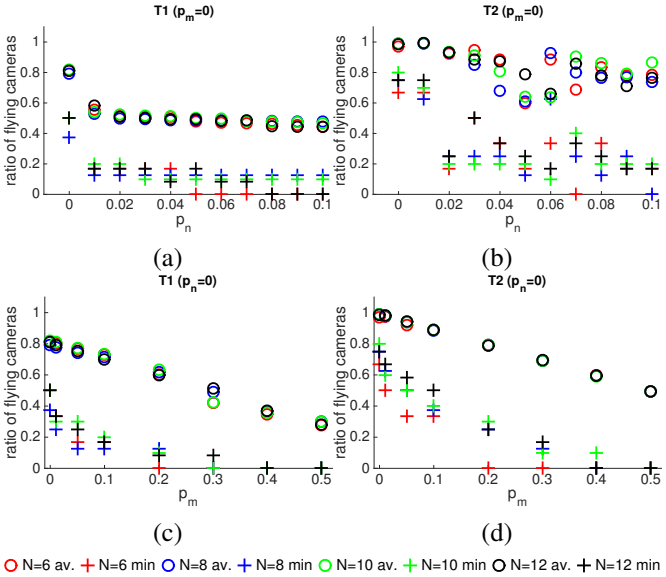

Fig. 7. Average (circle marker) and minimum (cross marker) ratios of flying cameras viewing the moving target over time with additive Gaussian noise on the target position and probability of missed detections on the camera plane of each camera. (a,b) The horizontal axis represents the variations of p_n's value. (c,d) The horizontal axis represents the variations of p_m's value.

cameras' trajectories. Fig. 8 shows also the points of view from the six flying cameras. These views are simulated by placing the cameras of the simulator in the position of the flying cameras and by approximating the angle of view ϕ.

Fig. 9 shows the average formation speed when changing initial velocities using 6 and 10 flying cameras (FC) on LT, ST, T1 and T2. Considered that on average $||v_t(0)|| \sim 2.5\Delta_k$, the initializations with $v_i(0) = (0,0,0)^T$ and $v_i(0) = (5\Delta_k, 0, 0)^T$ lead to either instability or incapability of following the target because the control for either small or large initial speeds cannot adapt to the speed of the target. This happens because we use only one mapping function to translate the target position on the camera plane to flight velocity adjustments. A possible solution to address this problem can be a multi-phase control where multiple mapping functions $M(k)$, with different parameters, are used based on the target position on the camera plane. In fact, the thrust to provide to the flying camera should be more sensitive at initialization and have a subsequent smoother reaction for steady target tracking.

Finally, we show qualitative results of an automatically generated video using the flying cameras' views deemed to be the best over time. The best cameras are selected by using a maximum-rank based strategy [25]. At a given k we select the flying camera whose viewed target is the closest to the center of the camera plane. We then maintain that view for a randomly-selected number of time steps $\tau_v = U[\tau_{v,min}\ \tau_{v,max}]$, where $\tau_{v,min}$ and $\tau_{v,max}$ are the minimum and maximum time to maintain each view, respectively. When τ_v is passed, we select the next best camera that is not the same, non-adjacent and within $135°$ from that just selected. Video results are available at `http://www.eecs.qmul.ac.uk/~andrea/flyingcameras.html`.

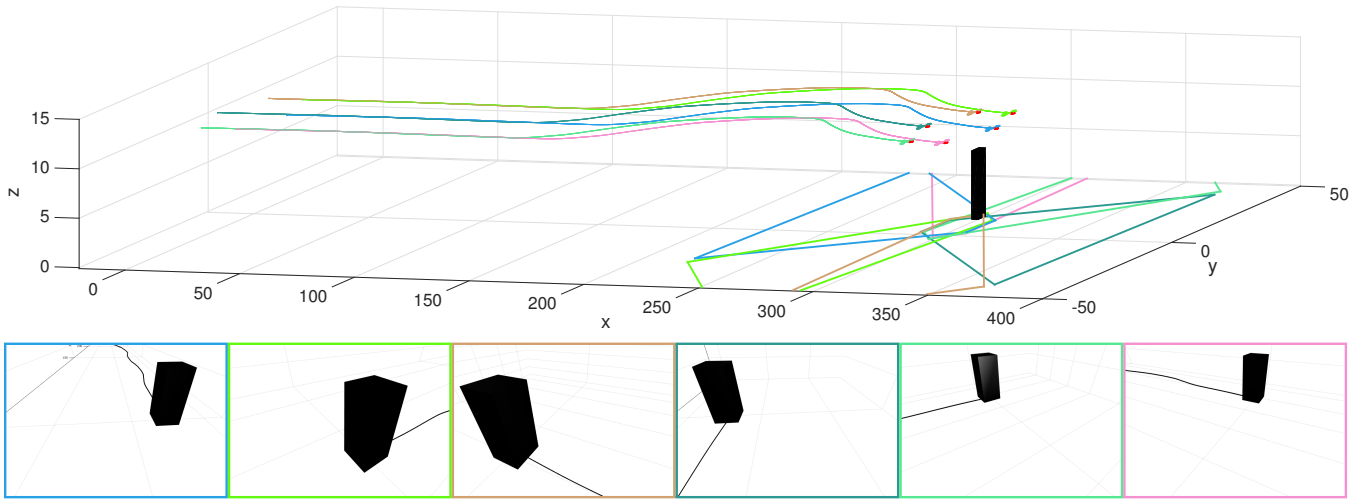

Fig. 8. Sample results from T2 at time instant 3500 from an overall view to individual views of each flying camera. The individual views from left to right correspond to the views from the front (light-blue) flying camera moving anti-clockwise.

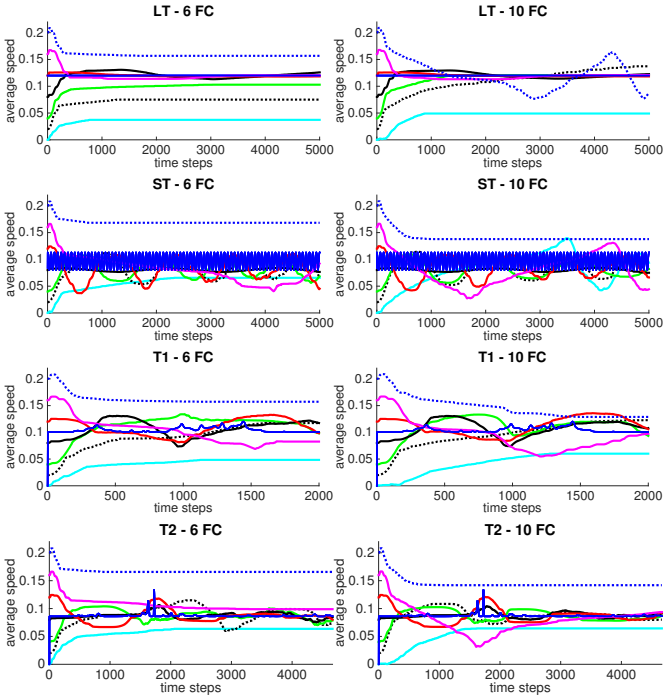

Fig. 9. Average formation speed with changing initial velocities using 6 and 10 flying cameras (FC) on LT, ST, T1 and T2. Lines: (cyan) $v_i(0) = (0,0,0)^T$; (dotted-black) $v_i(0) = (0.5\Delta_k, 0, 0)^T$; (green) $v_i(0) = (\Delta_k, 0, 0)^T$; (black) $v_i(0) = (2\Delta_k, 0, 0)^T$; (red) $v_i(0) = (3\Delta_k, 0, 0)^T$; (magenta) $v_i(0) = (4\Delta_k, 0, 0)^T$; (dotted-blue) $v_i(0) = (5\Delta_k, 0, 0)^T$. The blue line in the center is the speed of the target.

VII. Conclusions

We presented a vision-based thrust control approach that uses the target position on the camera plane to allow flying cameras to track a moving target. The target dynamics measured on the cameras plane are used to infer how and when to activate the adaptive thrust. We adopt a geometric-based algorithm to constrain the cameras to a fixed formation over time. The formation is preserved via a distributed algorithm that allows each camera to set a velocity computed with the knowledge of the velocity of its neighbors.

Our future research will involve the use of a PID controller to improve flight stability and to address the problem of flying at different altitudes especially to allow close-up views of the target. Also, it will be important to extend the method in the case of multiple targets and more complex target dynamics such as U-turns or abrupt accelerations.

References

[1] T. Akgun, C. Attwood, A. Cavallaro, C. Fabre, F. Poiesi, and P. Szczuko, "Towards cognitive and perceptive video systems," in *Human Behaviour Understanding in Networked Sensing*. Springer, 2014.

[2] D. Mellinger, N. Michael, and V. Kumar, "Trajectory generation and control for precise aggressive maneuvers with quadrotors," *Int. Jour. of Robotics Research*, vol. 31, no. 5, pp. 664–674, Apr. 2012.

[3] G. Vasarhelyi, C. Viragh, G. Somorjai, N. Tarcai, T. Szorenyi, T. Nepusz, and T. Vicsek, "Outdoor flocking and formation flight with autonomous aerial robots," in *Proc. of Intelligent Robots and Systems*, Chicago, USA, Sep. 2014.

[4] S. Shen, Y. Mulgaonkar, N. Michael, and V. Kumar, "Multi-sensor fusion for robust autonomous flight in indoor and outdoor environments with a rotorcraft MAV," in *Proc. of International Conference on Robotics and Automation*, Hong Kong, May 2014, pp. 4974–4981.

[5] J. Faigl, T. Krajnik, J. Chudoba, L. Preucil, and M. Saska, "Low-cost embedded system for relative localization in robotic swarms," in *Proc. of International Conference on Robotics and Automation*, Karlsruhe, DE, May 2013, pp. 993–998.

[6] T. Nageli, C. Conte, A. Domahidi, M. Morari, and O. Hilliges, "Environment-independent formation flight for micro aerial vehicles," in *Proc. of Intelligent Robots and Systems*, Chicago, USA, Sep. 2014.

[7] M. Aranda, G. Lopez-Nicolas, C. Sagues, and M.M. Zavlanos, "Three-dimensional multirobot formation control for target enclosing," in *Proc. of Intelligent Robots and Systems*, Chicago, USA, Sep. 2014.

[8] F. Poiesi and A. Cavallaro, "Self-positioning of a team of flying smart cameras," in *Proc. of Intelligent Sensors, Sensor Networks and Information Processing*, Singapore, Apr. 2015, pp. 1–6.

[9] J.F. Roberts, T. Stirling, J.-C. Zufferey, and D. Floreano, "3-D relative positioning sensor for indoor flying robots," *Autonomous Robots*, vol. 33, no. 1-2, pp. 5–20, Aug. 2012.

[10] F. Rivard, J. Bisson, F. Michaud, and D. Letourneau, "Ultrasonic relative positioning for multi-robot systems," in *Proc. of International Conference on Robotics and Automation*, Pasadena, USA, May 2008, pp. 323–328.

[11] B.D.O. Anderson, B. Fidan, C. Yu, and D. Walle, "UAV formation control: Theory and application," in *Recent Advances in Learning and Control*, vol. 371, pp. 15–33. Springer, 2008.

[12] J. Hu, L. Xie, J. Xu, and Z. Xu, "Multi-agent cooperative target search," *Sensors*, vol. 14, no. 6, pp. 9408–9428, Jun. 2014.

[13] M. Saska, J. Chudoba, L. Preucil, J. Thomas, G. Loianno, A. Tresnak, V. Von, and V. Kumar, "Autonomous deployment of swarms of micro-aerial vehicles in cooperative surveillance," in *Proc. of International Conference on Unmanned Aircraft Systems*, Orlando, USA, May 2014.

[14] M. Schwager, B.J. Julian, M. Angermann, and D. Rus, "Eyes in the sky: Decentralized control for the deployment of robotic camera networks," *Proc. of IEEE*, vol. 99, no. 9, pp. 1541–1561, Sep. 2011.

[15] L. Doitsidis, S. Weiss, A. Renzaglia, M.W. Achtelik, E. Kosmatopoulos, R. Siegwart, and D. Scaramuzza, "Optimal surveillance coverage for teams of micro aerial vehicles in GPS-denied environments using onboard vision," *Aut. Rob.*, vol. 33, no. 1-2, pp. 173–188, Aug. 2012.

[16] M. Blosch, S. Weiss, D. Scaramuzza, and R. Siegwart, "Vision based MAV navigation in unknown and unstructured environments," in *Proc. of Internation Conference on Robotics and Automation*, Anchorage, USA, May 2010, pp. 21–28.

[17] J. Portmann, S. Lynen, M. Chli, and R. Siegwart, "People detection and tracking from aerial thermal views," in *Proc. of International Conference on Robotics and Automation*, Hong Kong, Jun. 2014, pp. 1794–1800.

[18] P. Tokekar, V. Isler, and A. Franchi, "Multi-target visual tracking with aerial robots," in *Proc. of Intelligent Robots and Systems*, Chicago, USA, Sep. 2014, pp. 3067–3072.

[19] J. Valente, D. Sanz, J. Del Cerro, A. Barrientos, and M.A. de Frutos, "Near-optimal coverage trajectories for image mosaicing using a mini quad-rotor over irregular-shaped fields," *Precision Agriculture*, vol. 14, no. 1, pp. 115–132, Feb. 2013.

[20] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. of International Conference on Neural Networks*, Perth, AU, Nov. 1995, pp. 1942–1948.

[21] W. Kabsch, "A solution for the best rotation to relate two sets of vectors," *Acta Crystallographica Section A*, vol. 32, no. 5, pp. 922–923, Sep. 1976.

[22] T. Lee, M. Leok, and N.H. McClamroch, "Geometric tracking control of a quadrotor UAV on SE(3)," in *Proc. of International Conference on Robotics and Automation*, Atlanta, USA, Dec. 2010, pp. 5420–5425.

[23] X. Zhang, B. Xian, B. Zhao, and Y. Zhang, "Autonomous flight control of a nano quadrotor helicopter in a GPS-denied environment using on-board vision," *Trans. on Industrial Electronics*, (early access) 2015.

[24] M.H. Tanveer, S.F. Ahmed, D. Hazry, F.A. Warsi, and M.K. Joyo, "Stabilized controller design for attitude and altitude controlling of quad-rotor under disturbance and noisy conditions," *American Journal of Applied Sciences*, vol. 10, no. 8, pp. 819–831, 2013.

[25] F. Daniyal and A. Cavallaro, "Multi-camera scheduling for video production," in *Proc. of Visual Media Production*, London, UK, Nov. 2011, pp. 11–20.