Noname manuscript No.
(will be inserted by the editor)

Vision-based Drone Control for Autonomous UAV Cinematography

Ioannis Mademlis · Charalampos Symeonidis · Anastasios Tefas · Ioannis Pitas

Received: date / Accepted: date

Abstract One of the most important aesthetic concepts in autonomous Unmanned Aerial Vehicle (UAV) cinematography is the UAV/Camera Motion Type (CMT), describing the desired UAV trajectory relative to a (still or moving) physical target/subject being filmed. Usually, for the drone to autonomously execute such a CMT and capture the desired shot in footage, the 3D states (positions/poses within the world) of both the UAV/camera and the target are required as input. However, the target's 3D state is not typically known in non-staged settings. This paper proposes a novel framework for reformulating each desired CMT as a set of requirements that interrelate 2D visual information, UAV trajectory and camera orientation. Then, a set of CMT-specific vision-driven Proportional-Integral-Derivative (PID) UAV controllers can be implemented, by exploiting the above requirements to form suitable error signals. Such signals drive continuous adjustments to instant UAV motion parameters, separately at each captured video frame/time instance. The only inputs required for computing each error value are the current 2D pixel coordinates of the target's on-frame bounding box, detectable by an independent, off-the-shelf, real-time, deep neural 2D object detector/tracker vision subsystem. Importantly, neither UAV nor target 3D states are required ever to be known or estimated, while no depth maps, target 3D models or camera intrinsic parameters are necessary. The method was implemented and successfully evaluated in a robotics simulator, by properly reformulating a set of standard, formalized UAV CMTs.

Ioannis Mademlis E-mail: imademlis@csd.auth.gr · Charalampos Symeonidis E-mail: charsyme@csd.auth.gr · Anastasios Tefas E-mail: tefas@csd.auth.gr · Ioannis Pitas E-mail: pitas@csd.auth.gr
Department of Informatics, Aristotle University of Thessaloniki, Thessaloniki, Greece

Keywords UAV cinematography · intelligent shooting · autonomous drones · PID control · deep neural networks · vision-based control · mobile robotics · object detection

1 Introduction

The advent of camera-equipped, Vertical Take-Off and Landing (VTOL) Unmanned Aerial Vehicles (UAVs, or "drones") during the past decade has revolutionized the media industry, making the capture of impressive aerial footage easily accessible to almost anyone. Drones permit access to narrow spaces, novel visual effects, easy deployment, as well as rapid and flexible shot setup. In general, they are fully capable of replacing expensive helicopters, dollies and cranes at a low cost.

However, the logistics of employing manually operated drones in large-scale professional TV/film production are prohibitive, since at least two persons, i.e., a pilot and a camera operator, are required per UAV, having to cooperate in full coordination. This issue is aggravated when a swarm/fleet of multiple drones is employed for filming, due to the advantages multiple-UAV shooting offers: simultaneous capture of different targets, potential for more complex shots with multiple views, enhancement of the director's artistic palette and scene coverage [33][34][32].

Thus, overall, incorporating advanced autonomous functionalities to cinematography-oriented UAVs is highly desirable, so as to reduce the number of human operators and to alleviate their burden. Lying at the crossroad of aerial robotics, aerial cinematography, machine learning and computer vision, the emerging field of autonomous UAV cinematography attempts to develop robust robotic UAV platforms for intelligent filming [22] [40] [39] [7], that require minimal human intervention while obeying artistic guidelines. This line of research is complementary to automated

UAV video aesthetics evaluation [29]; the former refers to the media production stage, while the latter to the post-production stage.

One of the most important relevant cinematography concepts is the UAV/Camera Motion Type (CMT), describing the desired UAV trajectory relative to a (still or moving) physical target being filmed [6]. Also important is the Framing Shot Type (FST), which can be quantified as a target video frame area coverage percentage [35] [26] [48]. Examples of FSTs are the Close-Up, the Medium Close-Up, the Medium Shot, etc.

Typically, for the drone to autonomously execute a CMT and capture the desired shot in footage, the 3D position/poses of both the UAV/camera and the target are required as input; then, it is simply a matter of devising the error signal that drives appropriate trajectory tracking controllers operating in 3D space [36][1][2][10].

However, 3D position/pose is not typically known for the target in non-staged settings. In such a scenario, the most straightforward substitute is to rely on visual information and control the UAV in 3D space based on it. The traditional solution of estimating 3D target position/pose from visual data (e.g., using a Perspective-n-Point solver [15]) and employing this estimate for computing a control law is both error-prone and too complex, since it demands accurate camera calibration, known UAV state and, possibly, predefined visible landmarks on the target's image, along with their 3D correspondences in a target-based coordinate system.

The alternative solution is to rely on purely vision-driven controllers, where control laws are formulated directly on 2D image space, without attempting to estimate 3D state. Similar controllers have not yet been proposed for cinematography purposes, although they are nowadays common in more essential operations like landing [54]. Importantly, related existing approaches for active visual tracking do not consider the need for executing a specific CMT: the vehicle simply follows the moving visible target in 3D space, without regard for its own trajectory [13]. A multitude of similar vision-based active target tracking methods exist in the literature and rudimentary implementations can be found even on affordable commercial quadcopters, but such an approach does not take into account the aesthetic element of a desired CMT.

This paper presents a novel vision-based framework for designing Proportional-Integral-Derivative (PID) controllers that permit autonomous UAV CMT execution without relying on 3D state information. The proposed method involves reformulating any desired CMT as a set of requirements that interrelate 2D visual information, UAV trajectory and camera orientation. Then, a corresponding set of vision-driven PID controllers are designed and implemented by exploiting the above requirements to form suitable error signals, driving continuous adjustments to instant UAV motion parameters at each video frame/time instance. Although the details necessarily differ for each supported CMT, the general methodological framework is common.

The main novelty lies in the fact that *neither UAV nor target 3D states are required to be known or estimated in global coordinates*, thus making this method suitable for filming with inaccurate or intermittent Global Positioning System (GPS) signals. Additionally, in contrast to existing vision-based control approaches currently used in autonomous systems, *no depth maps, target 3D models or intrinsic camera parameters are needed*. The only inputs required are certain pieces of information which are always readily available from the on-board Inertial Measurement Units (IMUs), such as the relative angle between the camera axis and the UAV velocity vector at each time instance, as well as the on-frame bounding box of the visible target image in 2D pixel coordinates. This can be detected by an independent, real-time, off-the-shelf 2D object detector/tracker (e.g., a Deep Neural Network running on-board the vehicle), using simple RGB camera feed. This is significantly more robust and reliable than target 3D state estimation from visual data and camera parameters, while simultaneously foregoing any need for expensive sensors.

In short, the contributions of this paper are two-fold:

- A vision-based UAV/camera controller design framework is proposed, facilitating the construction of PID controllers that autonomously execute CMTs, based purely on what the drone-mounted camera "sees" in 2D pixel coordinates at each time instance, without the need for 3D state information. The required input may be extracted in real-time by off-the-shelf 2D object detector/trackers. Thus, existing computer vision solutions can be readily utilized for providing the proper PID error signal at each time instance.
- The proposed method is concretely implemented for a set of industry-standard, formalized UAV cinematography CMTs, extracted from a recently proposed UAV shot type taxonomy [35] [32] [24] [25] [26]. The derived controllers, serving as detailed examples of the proposed method, are extensively evaluated in simulation.

This paper focuses on UAV cinematography CMTs due to their high industry relevance for media production. However, the proposed method is actually more generic: it can be applied to any setting where a specific UAV motion trajectory relative to a visible target/subject is desired. For instance, travelling to a detectable landing spot in order to dock is typically performed by flying horizontally (at a constant altitude) towards the landing location, until the UAV hovers directly above it, and subsequently flying down at a purely vertical direction. This is a trajectory that can be modelled by the proposed framework as well as the example cinematography CMTs.

2 Related Work

When deploying camera-equipped UAVs, if it is not necessary for the vehicle to execute a specific trajectory, but simply to keep following and filming a specific moving target, then active visual tracking approaches may be utilized [13] [60]. Rudimentary methods of this type can indeed be found in almost all commercial multicopters with target tracking capabilities. However, this is a significantly limited version of the UAV motion control problem and does not really apply to professional cinematography scenarios, where the camera trajectory is actually an essential aesthetic element in itself.

Thus, in autonomous UAV cinematography applications, motion control typically involves trajectory tracking controllers that operate in 3D space and rely on GPS/IMU signal fusion [16] [38]. Assuming the target's 3D state is also known, the appropriate trajectory itself can be planned according to the desired CMT specified by the director (e.g., using the CMT geometrical modelling in [35]). In general, Proportional-Integral-Derivative (PID) [57] or Linear-Quadratic Regulator (LQR) [41] controllers are employed for related tasks. The PixHawk/PX4 Autopilot [37], a popular low-level flight trajectory control system, offers a commercial off-the-shelf PID cascade control solution for UAVs that allows vehicle steering at various levels, ranging from designating 3D path waypoints to directly feeding raw motion commands to the motors.

When the target's 3D state is unknown, its estimation from visual data is a feasible substitute; then traditional controllers with error signals defined in 3D space may be employed. Although Visual SLAM [50] cannot be readily employed if the target is moving, target 3D position/pose may be estimated from the UAV's camera feed using a Perspective-n-Point problem solver, which requires accurate camera calibration and predefined visible 2D landmarks on the target's image, along with their 3D correspondences in a target-based coordinate system. Alternatively, mostly in the case of human targets, Deep Neural Networks (DNNs) can be employed for detecting the subjects and estimating their 2D pose on-frame [51] [44] [45]. Their output can then be combined with known UAV 3D state/camera parameters to acquire a complete estimate of target 3D state. However, DNNs such as Convolutional Neural Networks (CNNs) or Long Short-Term Memory architectures (LSTMs) that can be utilized for extracting 3D geometry have not yet reached the level of maturity required by robotics applications operating in the physical world, due to the commonly arising issue of domain shift distributions between the training set and the inference-stage input data. In general, these solutions are not robust, reliable or flexible enough, while accurate knowledge of the UAV's 3D state is typically still demanded.

The alternative solution is to rely on purely vision-based controllers, where control laws are formulated directly on 2D image space, without attempting to estimate 3D state. This is typically called Image-Based Visual Servoing (IBVS) [58], where the goal is to control 3D camera velocity based on the current deviation of the positions (in 2D pixel coordinates) of on-target landmark point images from prespecified desired ones. Typically, IBVS requires knowledge of camera intrinsic parameters and a depth map per video frame, at least in principle. Similar controllers have not yet been proposed for cinematography purposes, although they are nowadays common in more essential operations like safe landing [54] [23]. In similar scenarios, the camera orientation/position on the vehicle may be considered fixed, so that 3D UAV velocity is the quantity in need of control.

The degree of precision required in autonomous UAV cinematography is not as great as, e.g., in applications involving object manipulation by robotic grippers. This fact has recently been exploited in [47] to design a simple camera/gimbal PID controller for cinematography purposes, that does not depend either on UAV or target 3D state knowledge or estimate. The controller is able to physically rotate a gimbal so as to continuously keep a desired target's 2D image positioned (in pixel coordinates) at a user-defined prespecified image region (e.g., video frame center) and constantly adjust camera zoom level to maintain the desired FST at all times [35]. The only input required is a rectangular target 2D Region-of-Interest per video frame (also in pixel coordinates), which can be reliably extracted in real-time by modern deep neural, single-stage object detectors and trackers [12] [42][56][49]. This is a task that DNNs can perform much more reliably in the real world, compared to 3D geometry extraction. Although it still requires significant computational power in the form of embedded AI platforms designed for drones (e.g., the nVidia Jetson Tegra line of products), it is much more robust than 3D target state estimation and does not depend on known camera parameters or depth maps. However, the proposed controller does not consider CMTs at all, despite their nature as a cornerstone of UAV cinematography.

An alternative both to IBVS and to the simple vision-based PID control scheme from [47] is reinforcement learning (RL) employing raw video input and motor command output, which also discards any need for UAV or target 3D state knowledge or estimate. DNNs have recently been employed in similar settings for UAV collision avoidance [53], indoor flight control in search and recovery operations [28] or high-level flight navigation [27] [11]. An imitation learning variant has also been explored for drone racing [31], where a neural network learns to map video input to proper motor control commands in a supervised setting, using datasets obtained by employing human pilots in a photorealistic simulator. More generally, such approaches rely on advanced robotics simulators where the training of RL agents takes place; successful generalization to unknown real-world environments during actual deployment is not guaranteed. This disparity between training conditions in the simulator and inference conditions in the real world is known as the "reality gap".

Additionally, RL has not yet been investigated for cinematography applications, with the exceptions of [46] and [17]. In the first one, the task is not to autonomously execute specific CMTs, but simply to capture frontal close-up shots of human targets. In the second one, the task does not involve low-level UAV control for CMT execution, but rather fully autonomous high-level, on-line cinematography planning that draws from a limited palette of rudimentary CMTs.

A different, non-RL DNN architecture for learning UAV cinematography in a supervised manner is explored in [18], using real, professional UAV video footage, where a Sequence-to-Sequence Convolutional LSTM learns to regress desired dense optical flows (OFs) from input dense OFs and CNN-derived semantic features extracted by each video frame (describing target on-frame 2D position/pose and visible scene objects). The output OFs are analytically translated into desired UAV/camera motion commands using known camera parameters and an estimated Essential Matrix. Finally, the predicted motion commands over N immediate future timesteps are assembled into feasible UAV trajectories by exploiting any trajectory planning method.

In a similar, but less generic algorithm [19], an LSTM-based encoder-decoder neural architecture for temporal Sequence-to-Sequence prediction tasks (e.g., neural machine translation) learns to regress the desired next filming target appearance, at each time instance, from purely visual inputs under mild assumptions (the filming target is a single subject of known height in 3D world). Training also takes place on professional UAV footage, rendering this too a variant of imitation learning. During inference, the DNN prediction is translated into UAV/camera motion commands using a PID controller and a trajectory planning algorithm. The underlying neural building blocks are rather straightforward: two LSTM networks and an attention layer [5].

Both of these imitation learning approaches, although not suffering from the reality gap typical of RL-based methods, are constrained by their reliance on a specific training dataset and their lack of explicit CMT modelling. In contrast, in this paper, a generic method is proposed for designing specific, purely vision-driven PID controllers per CMT, while concrete examples are provided for a range of cinematography CMTs. To the best of our knowledge, *this is the first systematic presentation of vision-driven, stateless UAV controllers for autonomous CMT execution that operate without knowing or estimating the filming target's 3D state; we are not aware of any similar, published algorithms*.

By not relying on machine learning they do not require training, either on a professional UAV footage dataset or on a simulator, thus they do not suffer as much in case of a discrepancy between development and deployment conditions. By not relying on known or estimated 3D information, since the derived PID controllers are only driven by an error signal computed on 2D image space, they achieve high robustness even in flight environments with intermittent and/or inaccurate GPS signals. Finally, in contrast to typical IBVS solutions for autonomous systems, the proposed framework does not require neither depth maps, nor knowledge of intrinsic camera parameters.

3 Vision-based UAV Control for Autonomous CMT execution

The proposed method is, in essence, a methodological framework for designing PID controllers that allow a UAV to autonomously execute a specific CMT, using only 2D visual input. Importantly, a different set of dedicated controllers must be designed for each CMT.

3.1 Proposed Method

A target-tracking CMT can be defined as a set of requirements that interrelate captured 2D visual information and parameters of UAV trajectory and camera orientation. These requirements can then be exploited to form error signals, driving corresponding PID controllers that manipulate instant UAV/camera motion parameters. Thus, a target-tracking CMT can be executed without needing 3D target/UAV coordinates, relying only on what the UAV "sees" at each time instance. The only requirement is for the target to initially be visible on the first video frame. Subsequently, a detected target ROI (in 2D pixel coordinates) is assumed to be available at each video frame using relevant automated, off-the-shelf algorithms.

Designing a set of CMT-specific PID controllers according to the proposed method involves three sequential steps:

1. Identify relevant parameters and external values.
2. Identify relevant controllers.
3. Specify the proper error signal (including a reference value) for each required controller.

These stages are described below in detail.

The first step consists in defining the desired CMT as a set of requirements, specifying at each time instance the valid value range of a *subset* of the 12 UAV/camera/target ROI parameters shown in Table 1. Not all of the above parameters are relevant to each CMT. Depending on the specific CMT, a number of these parameters must stay fixed during execution (to pre-identified "external values"), while

Table 1: Complete set of required UAV/camera/target ROI parameters.

1.	Scalar UAV speed v
2.	Scalar UAV speed along the vertical axis v_v
3.	Scalar UAV speed along the horizontal axis v_h
4.	Absolute pitch rotation angle θ_D of the UAV 3D velocity vector
5.	Relative yaw angle ω_ϕ between the UAV 3D velocity direction and the 3D camera axis
6.	Relative pitch angle ω_θ between the UAV 3D velocity direction and the 3D camera axis
7.	Absolute pitch angle θ_C of the 3D camera axis
8.	Absolute yaw angle ϕ_C of the 3D camera axis
9.	Pitch angular velocity a of the 3D camera axis
10.	Target 2D ROI area R_a
11.	Target 2D ROI center $[R_x, R_y]^T$ (in pixel coordinates)
12.	Scalar camera focal length f

Table 2: Complete set of controlled parameters.

1.	UAV speed v
2.	Instant UAV speed along the horizontal axis Δv_h
3.	Instant UAV speed along the vertical axis Δv_v
4.	Instant camera pitch rotation angle $\Delta\theta_C$
5.	Instant UAV pitch rotation angle $\Delta\theta_D$
6.	Instant UAV yaw rotation angle $\Delta\phi_D$
7.	Camera focal length f

others must be implicitly controlled by the PID controllers constructed according to the proposed method.

Thus, the second step consists in identifying the N controllers, where $N \in \mathbb{N}, 1 \leq N \leq 7$, that are required for the desired CMT. These will control a *subset* of the 7 scalar parameters shown in Table 2. As before, not all of these parameters are relevant to each specific CMT.

Finally, the third step consists in expressing the scalar, time-dependent error signal $e(t)_i$ for the i-th PID controller, where $1 \leq i \leq N$. This is based on the current deviation of a function of the identified parameters from a “reference value”, at each time instance/video frame t.

Subsequently, during actual operation, the overall error signal $\mathbf{e}(t) \in \mathbb{R}^N$ (where N is the number of PID controllers required for specifying the desired CMT) drives the output of the well-known PID equation [4]:

$$\mathbf{u}(t) = \mathbf{K}_p\mathbf{e}(t) + \mathbf{K}_i \int_0^\tau \mathbf{e}(t)\, d\tau + \mathbf{K}_d \frac{d\mathbf{e}(t)}{dt}, \tag{1}$$

where all the operators are applied element-wise and $\mathbf{K}_p \in \mathbb{R}^N$, $\mathbf{K}_i \in \mathbb{R}^N$ and $\mathbf{K}_d \in \mathbb{R}^N$ are the coefficients of the proportional, integral and derivative terms. The output $\mathbf{u}(t) \in \mathbb{R}^N$ is a vector containing the control commands for the t-th time instance/video frame.

Both the external and the reference values must have been prespecified separately for each CMT. The choice of certain external values affects aesthetic properties of the captured footage (e.g., its FST, or the shot duration), thus there is no single “correct value” for them, while for others a specific value (or range of values) must have been selected so that the CMT can be described properly. Overall, a consistent and valid set of parameters, external values, reference values, controllers and corresponding error signals jointly describes a given CMT, in a manner leading to its autonomous execution. Absolute positions in 3D space (either of the UAV or the target), intrinsic camera parameters or depth maps are not required: they do not have to be known, estimated or provided.

For simplicity, we assume that controllers developed according to the proposed method send instantaneous control commands to UAV/camera parameters in discrete time instances coinciding with discrete, incoming video frames. In practice, an approximately fixed lag (in the order of a few msecs) may intervene between capturing a video frame and sending the corresponding control commands, due to 2D visual target detection/tracking requirements. However, this lag is typically negligible for UAVs properly equipped with modern real-time algorithms and computational hardware.

3.2 PID Controllers for Autonomous UAV Cinematography

To showcase the proposed method, we concretely implemented it for a set of industry-standard, formalized UAV cinematography CMTs, extracted from a recently proposed UAV shot type taxonomy [35] [32] [24] [25] [26]. Figure 1 graphically summarizes the supported CMTs, while brief textual descriptions can be found in Table 3. More details are in [35].

For simplicity, we assume central composition and that all pixel coordinates have been rescaled into the interval $[0, 1]$, although these are not strictly necessary. The supported CMTs can be divided into ones that inherently retain constant FST and ones that do not (ORBIT, FLYBY, FLYOVER, DESCENT, ASCENT, PST, CONLTS). In the second case, a controller that continuously adjusts camera focal length f and the corresponding component of the error signal are foreseen, in order to also retain constant FST, besides implementing the CMT. However, this controller/error component is entirely optional and can be removed (thus, fixing f to a stable value), if it is not desired to maintain a specific FST while capturing the shot. If it is indeed employed, however, it can operate without requiring knowledge of the actual camera focal length at each time instance; only the firmware/hardware ability to slightly increase or decrease f with a single command is needed.

Below, the terms "local tangent plane" and "terrain tangent plane" refer to a plane parallel to the local sea level and to the plane instantaneously tangent to the local terrain inclination, respectively. The formulas and notations have been derived by assuming an East-North-Up (ENU) [14] 3D coordinate system and OpenCV [8] 2D coordinate conventions. In the camera reference frame C, the x-axis is aligned with the camera axis, while the z-axis is pointing up. The absolute reference frame W is defined at $t = 0$ (exactly before each CMT execution), where the target is visible in the video frame, using the conventions below:

- The y-axis of W is aligned with the y-axis of C.
- The z-axis of W is perpendicular to the local tangent plane.

In the sequel, the set of controllers implementing each of the supported cinematography CMTs according to the proposed method is described in detail.

3.2.1 ORBIT

Orbit (ORBIT) can be described as the following set of requirements:

- The UAV should retain constant speed v (greater than target speed u). This is an external value that implicitly determines the drone's angular velocity.
- The UAV velocity direction and the camera axis should form and retain a yaw angle of ω_ϕ equal to $\pm\frac{\pi}{2}$ rad. This is initially controlled/enforced, by fixing the camera axis yaw rotation angle and properly modifying the UAV yaw rotation angle at each time instance, via the corresponding PID controller (detailed below). Once the proper ω_ϕ has been achieved, it is locked by compensating each change in UAV yaw rotation angle with a corresponding change in the camera axis yaw rotation angle.
- The camera axis pitch angle should remain fixed to $\theta_C > 0$. This is an external value that implicitly determines the UAV altitude relative to the target.
- Target 2D ROI area R_a should remain approximately fixed to d_a. d_a is an external value that implicitly determines the desired FST.
- Target 2D ROI center $[R_x, R_y]^T$ should always remain at the video frame center/principal point.

The following controllers must be constructed, based on the above requirements:

- Instant UAV yaw rotation angle $\Delta\phi_D$.
- Instant UAV pitch rotation angle $\Delta\theta_D$.
- The camera focal length f.

Therefore, the ORBIT error signal can be written as follows:

$$\begin{aligned}\mathbf{e}(t) = [&R_x - 0.5 + \operatorname{sgn}(\sin\omega_\phi)\cos\omega_\phi, \\ &0.5 - R_y, R_a - d_a]^T \in \mathbb{R}^3.\end{aligned} \tag{2}$$

By plugging this error signal into Eq. (1), $u(t)_i$ ends up depending on $e(t)_i, 1 \leq i \leq 3$. Thus, ultimately, $u(t)_1/u(t)_2/u(t)_3$ is the control command sent by the $\Delta\phi_D/\Delta\theta_D/f$ controller at the t-th time instance, respectively. Overall, given that the UAV retains a constant scalar linear speed greater than the target speed, Eq. (2) penalizes the horizontal/vertical distance of the target ROI center from the image center, in order to enforce the desired circular UAV motion around the target and maintain central composition. Also, it uses the sgn function to determine whether the target lies to the left or to the right side of the UAV. Current $\cos\omega_\phi$ at each time instance is exploited in order to push ω_ϕ towards the desired right angle and keep it fixed there, while the f controller relies on an error based on the difference between the desired and the actual target 2D ROI area.

The proposed error signals and the control outputs for the remaining CMTs, which are detailed below, were derived in a similar manner.

3.2.2 CHASE

Chase (CHASE) can be described as the following set of requirements:

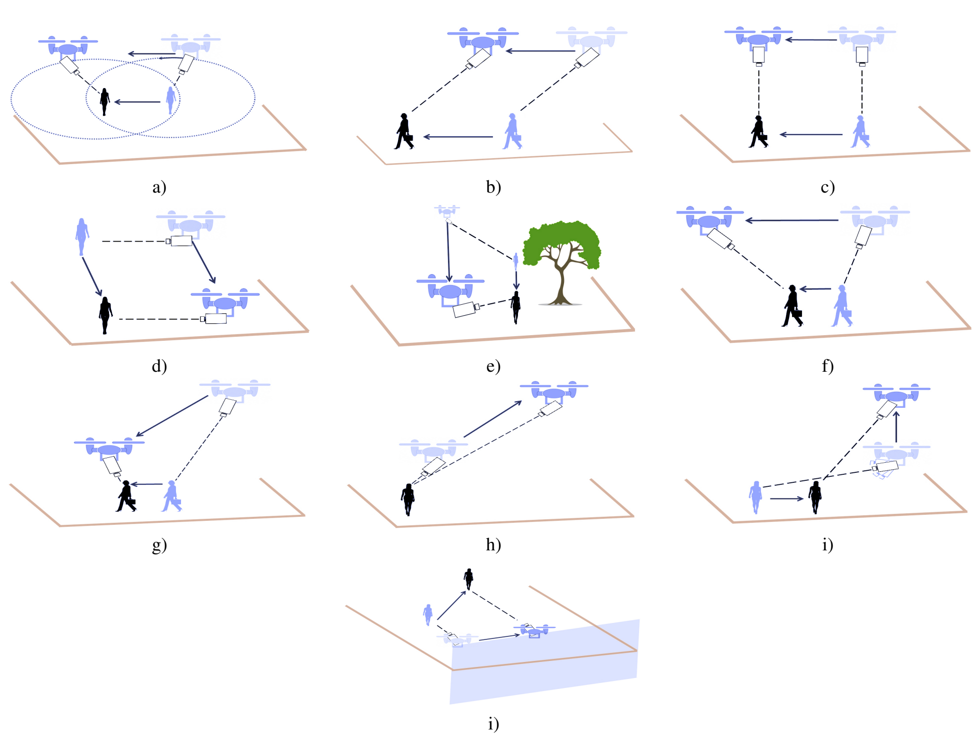


Fig. 1: Graphical illustration of the supported UAV CMTs: a) ORBIT, b) CHASE, c) VTS, d) LTS, e) FLYBY, f) FLYOVER, g) DESCENT, h) ASCENT, j) PST and k) CONLTS.

– The UAV velocity direction and the camera axis should always form and retain a yaw angle ω_ϕ equal either to 0 or π rad. This is initially controlled/enforced, by fixing the camera axis yaw rotation angle and properly modifying the UAV yaw rotation angle at each time instance, via the corresponding PID controller (detailed below). Once the proper ω_ϕ has been achieved, it is locked by compensating each change in UAV yaw rotation angle with a corresponding change in the camera axis yaw rotation angle.
– The camera axis pitch angle should remain fixed to $\theta_C > 0$. This is an external value that implicitly determines the UAV altitude relative to the target.
– Camera focal length f should remain constant.
– Target 2D ROI area R_a should remain approximately fixed to d_a. d_a is an external value that implicitly determines the desired FST.
– Target ROI center $[R_x, R_y]^T$ should always remain at the video frame center/principal point.

The following controllers must be constructed, based on the above requirements:

– UAV speed v.
– Instant UAV yaw rotation angle $\Delta\phi_D$.
– Instant UAV pitch rotation angle $\Delta\theta_D$.

Therefore, the CHASE error signal can be written as follows:

$$\begin{aligned}\mathbf{e}(t) = [&(R_a - d_a)\,\mathrm{sgn}\,(\cos\omega_\phi), \\ &(R_x - 0.5)\,\mathrm{sgn}\,(\cos\omega_\phi) - \mathrm{sgn}\,(\cos\omega_\phi)\sin\omega_\phi, \\ &0.5 - R_y]^T \in \mathbb{R}^3.\end{aligned} \tag{3}$$

3.2.3 VTS

Vertical Tracking Shot (VTS) can be described as the following set of requirements:

– The UAV velocity direction and the camera axis should always form and retain a yaw angle ω_ϕ equal to 0 rad. This is initially controlled/enforced, by fixing the camera axis yaw rotation angle and properly modifying the UAV yaw rotation angle at each time instance, via the corresponding PID controller (detailed below). Once the proper ω_ϕ has been achieved, it is locked by compensating each change in UAV yaw rotation angle with a corresponding change in the camera axis yaw rotation angle.
– The UAV velocity direction and the camera axis should always form and retain a pitch angle ω_θ equal to $\frac{\pi}{2}$ rad. This can be either expressly controlled/enforced by the proposed algorithm, using $\cos\omega_\theta$ as an error signal, or be a fixed external value that defines the VTS CMT.

Table 3: Textual description of the supported UAV CMTs [35].

CMT Name	CMT Description
ORBIT	The camera gimbal is constantly rotating, so as to always keep the still or linearly moving target properly framed. The UAV circles around and above the target, while simultaneously following the latter's linear trajectory (if it is not still). During shooting, the UAV altitude varies according to the target altitude.
CHASE	The camera gimbal remains stable and the camera always points at the moving target. The UAV follows/leads the target from behind/from the front at a steady distance, by matching its speed, if possible.
VTS	The camera gimbal remains stable and the camera always points at the moving target. The camera axis is always perpendicular to the target velocity vector and the UAV flies exactly above the target, matching its speed, if possible.
LTS	The camera gimbal remains stable and the camera always focused on the moving target. In LTS, the camera axis is approximately perpendicular both to the target velocity vector and to the terrain tangent plane normal vector, while the UAV flies sideways/in parallel to the target, matching its speed, if possible.
FLYBY	The camera gimbal is constantly rotating, so as to always keep the still or linearly moving target properly framed. The UAV intercepts the target from behind/from the front and to the left/right, maintaining steady vertical distance and constant velocity, passes it by and keeps on flying at a linear trajectory, with the camera still pointing at the receding target. The UAV and target velocity vector projections onto the local tangent plane remain approximately parallel during shooting.
FLYOVER	The camera gimbal is constantly rotating along the pitch axis, so as to always keep the still or linearly moving target properly framed. The UAV intercepts the target from behind/from the front, maintaining a steady vertical distance and constant velocity, flies exactly above it and keeps on flying at a linear trajectory, with the camera still pointing at the receding target. The UAV and target velocity vector projections onto the local tangent plane remain approximately parallel during shooting.
DESCENT	The camera gimbal may constantly rotate along the pitch axis, so as to always keep the linearly moving target properly framed. Gimbal rotation is not necessary in the case of a still target. The UAV flies linearly and intercepts the target from behind or from the front, at a steadily decreasing vertical distance and constant velocity. If the target is still, the shot ends with the UAV flying exactly above it. The UAV and target velocity vector projections onto the local tangent plane remain approximately parallel during shooting.
ASCENT	The camera gimbal may constantly rotate along the pitch axis, so as to always keep the linearly moving target properly framed. Gimbal rotation is not necessary in the case of a still target. The UAV flies linearly, backing away from the target, at a steadily increasing vertical distance and with constant velocity. The UAV and target velocity vector projections onto the local tangent plane remain approximately parallel during shooting.
PST	The UAV is slowly flying up or down, along the k-axis, with constant velocity. The camera gimbal rotates slowly (mainly along the pitch axis), so as to always keep the still or linearly moving target properly framed. The projections of the camera axis and the target trajectory on the ground plane are approximately lying on the same line during shooting.
CONLTS	The camera gimbal remains stable and the camera always points at the moving target. The UAV flies along the projection of the target trajectory onto a predefined "flight plane", vertical to the ground plane, while maintaining a constant altitude relative to the target during shooting. This is useful, for instance, in football match coverage, where the UAVs are allowed to fly only above the pitch sidelines.

– Camera focal length f should remain constant. This is an external value that implicitly determines the UAV altitude relative to the target.
– Target 2D ROI area R_a should remain approximately fixed to d_a. d_a is an external value that implicitly determines the desired FST.
– Target ROI center $[R_x, R_y]^T$ should always remain at the video frame center/principal point.

The following controllers must be constructed, based on the above requirements:

– UAV speed v.
– Instant UAV yaw rotation angle $\varDelta\phi_D$.
– Instant UAV pitch rotation angle $\varDelta\theta_D$.

Therefore, the VTS error signal can be written as follows:

$$\mathbf{e}(t) = [R_y - 0.5, R_x - 0.5 - \sin\omega_\phi, \\ R_a - d_a]^T \in \mathbb{R}^3. \quad (4)$$

3.2.4 LTS

Lateral Tracking Shot (LTS) can be described as the following set of requirements:

– The UAV velocity direction and the camera axis should always form and retain a yaw angle of $\omega_\phi = \pm\frac{\pi}{2}$ rad. This is initially controlled/enforced, by fixing the camera axis yaw rotation angle and properly modifying the UAV yaw rotation angle at each time instance, via the corresponding PID controller (detailed below). Once the proper ω_ϕ has been achieved, it is locked by compensating each change in UAV yaw rotation angle with a corresponding change in the camera axis yaw rotation angle.
– The camera axis pitch angle should remain fixed to $\theta_C \leq 0$. This is an external value that implicitly determines the UAV altitude relative to the target. In an ideal LTS shot (rarely feasible), it holds that $\theta_C = 0$.

- Camera focal length f should remain constant. This is an external value that implicitly determines the horizontal UAV-target distance.
- Target 2D ROI area R_a should remain approximately fixed to d_a. d_a is an external value that implicitly determines the desired FST.
- Target ROI center $[R_x, R_y]^T$ should always remain at the video frame center/principal point.

The following controllers must be constructed, based on the above requirements:

- UAV speed v.
- Instant UAV yaw rotation angle $\Delta\phi_D$.
- Instant UAV pitch rotation angle $\Delta\theta_D$.

Therefore, the LTS error signal can be written as follows:

$$\mathbf{e}(t) = [bc, (R_a - d_a)c + c\cos\omega_\phi + H(d_a - R_a)\,\alpha\,|b|\,c, \\ 0.5 - R_y]^T \in \mathbb{R}^3, \tag{5}$$

where H is the Heaviside step function, α is an empirically set small positive coefficient and:

$$b = 0.5 - R_x, \tag{6}$$

$$c = \operatorname{sgn}(\sin\omega_\phi). \tag{7}$$

The purpose of the term $H(d_a - R_a)\,\alpha\,|b|\,c$ is to reduce the magnitude of the perceived error in 2D ROI size in cases where the ROI is not located at the video frame center. Thus, when the UAV is either in front of or behind the target along the latter's 3D trajectory and, consequently, the ROI area is shrinked due to the increased 3D UAV-to-target distance, we avoid inappropriate control commands attempting to increase instant UAV yaw rotation angle. By adding this term to the controller error, contractions in 2D ROI size correctly affect the UAV yaw rotation angle only when the target turns left or right along its route in the 3D world. During the evaluation process described in Section 4, a suitable $\alpha = 0.015$ was empirically found.

3.2.5 FLYBY

Fly-By (FLYBY) can be described in two stages. In the first stage, the following requirements must be met:

- The camera axis retains a constant yaw angular velocity a, with the yaw angle ω_ϕ formed by the UAV velocity direction and the camera axis steadily increasing from 0 up to $\tilde{\omega}_\phi = \pm\frac{\pi}{4}$ rad. a is an external value that implicitly determines the shot duration.
- The camera axis pitch angle should remain fixed to $\theta_C > 0$. This is an external value that implicitly determines the UAV altitude relative to the target.
- Target 2D ROI area R_a should remain approximately fixed to d_a. d_a is an external value that implicitly determines the desired FST.
- Target ROI center $[R_x, R_y]^T$ should always remain at the video frame center/principal point.

The following controllers must be constructed, based on the above requirements:

- Instant UAV yaw rotation angle $\Delta\phi_D$.
- Instant UAV pitch rotation angle $\Delta\theta_D$.
- The camera focal length f.

In the second stage, the yaw angle ω_ϕ steadily increases/decreases from $\tilde{\omega}_\phi = \pm\frac{\pi}{4}$ up to $\pm\pi$ rad. The following controllers must be constructed:

- UAV speed v.
- Instant UAV pitch rotation angle $\Delta\theta_D$.
- The camera focal length f.

In the first stage, the FLYBY error signal can be written as follows:

$$\mathbf{e}(t) = [R_x - 0.5, \\ 0.5 - R_y, R_a - d_a]^T \in \mathbb{R}^3. \tag{8}$$

The corresponding error signal for the second stage is:

$$\mathbf{e}(t) = [(0.5 - R_x)\operatorname{sgn}(\sin\omega_\phi), \\ 0.5 - R_y, R_a - d_a]^T \in \mathbb{R}^3. \tag{9}$$

In-between the two stages, a short-duration intermediate phase is required where the yaw angle ω_ϕ remains fixed to $\tilde{\omega}_\phi = \pm\frac{\pi}{4}$ and the first-stage controllers keep running. The disadvantage of the overall formulation is that, in the case of a linearly moving target, it requires favourable initial UAV yaw orientation ϕ_D relative to the target 3D trajectory. Otherwise the UAV and target velocity vector projections onto the local tangent plane will not be approximately parallel during shot execution. Obviously, there is no such issue in the case of a still target.

3.2.6 FLYOVER

Fly-Over (FLYOVER) can be described as the following set of requirements:

- The camera axis retains a constant pitch angular velocity a, with the pitch angle ω_θ formed by the UAV velocity direction and the camera axis steadily increasing from 0 up to π rad. a is an external value that implicitly determines the shot duration.
- The UAV velocity direction and the camera axis should always form a yaw angle ω_ϕ equal to 0 rad. This is initially controlled/enforced, by fixing the camera axis yaw rotation angle and properly modifying the UAV yaw rotation angle at each time instance, via the corresponding

PID controller (detailed below). Once the proper ω_ϕ has been achieved, it is locked by compensating each change in UAV yaw rotation angle with a corresponding change in the camera axis yaw rotation angle.
- Target 2D ROI area R_a should remain approximately fixed to d_a. d_a is an external value that implicitly determines the desired FST.
- Target ROI center $[R_x, R_y]^T$ should always remain at the video frame center/principal point.

The following controllers must be constructed, based on the above requirements:

- UAV speed v.
- Instant UAV yaw rotation angle $\Delta\phi_D$.
- The camera focal length f.

Therefore, the FLYOVER error signal can be written as follows:

$$\mathbf{e}(t) = [R_y - 0.5, R_x - 0.5 - \sin\omega_\phi, R_a - d_a]^T \in \mathbb{R}^3. \quad (10)$$

3.2.7 DESCENT

Descent (DESCENT) can be described as the following set of requirements:

- The UAV should retain constant speed v (greater than target speed u). This is an external value that implicitly determines shot duration.
- The UAV should retain constant pitch rotation angle $\theta_D > 0$.
- The UAV velocity direction and the camera axis should always form a yaw angle ω_ϕ equal to 0 rad. This is initially controlled/enforced, by fixing the camera axis yaw rotation angle and properly modifying the UAV yaw rotation angle at each time instance, via the corresponding PID controller (detailed below). Once the proper ω_ϕ has been achieved, it is locked by compensating each change in UAV yaw rotation angle with a corresponding change in the camera axis yaw rotation angle.
- Target 2D ROI area R_a should remain approximately fixed to d_a. d_a is an external value that implicitly determines the desired FST.
- Target ROI center $[R_x, R_y]^T$ should always remain at the video frame center/principal point.

The following controllers must be constructed, based on the above requirements:

- Instant camera pitch rotation angle $\Delta\theta_C$.
- Instant UAV yaw rotation angle $\Delta\phi_D$.
- The camera focal length f.

Therefore, the DESCENT error signal can be written as follows:

$$\mathbf{e}(t) = [0.5 - R_y, R_x - 0.5 - \sin\omega_\phi, R_a - d_a]^T \in \mathbb{R}^3. \quad (11)$$

3.2.8 ASCENT

Ascent (ASCENT) can be described as the following set of requirements:

- The UAV should retain constant speed v. This should always be lower/greater than target speed u if the vehicle is behind/in front of the target, respectively.
- The UAV should retain constant pitch rotation angle $\theta_D < 0$.
- The UAV velocity direction and the camera axis should always form a yaw angle ω_ϕ equal to π rad. This is initially controlled/enforced, by fixing the camera axis yaw rotation angle and properly modifying the UAV yaw rotation angle at each time instance, via the corresponding PID controller (detailed below). Once the proper ω_ϕ has been achieved, it is locked by compensating each change in UAV yaw rotation angle with a corresponding change in the camera axis yaw rotation angle.
- Target 2D ROI area R_a should remain approximately fixed to d_a. d_a is an external value that implicitly determines the desired FST.
- Target ROI center $[R_x, R_y]^T$ should always remain at the video frame center/principal point.

The following controllers must be constructed, based on the above requirements:

- Instant camera pitch rotation angle $\Delta\theta_C$.
- Instant UAV yaw rotation angle $\Delta\phi_D$.
- The camera focal length f.

Therefore, the ASCENT error signal can be written as follows:

$$\mathbf{e}(t) = [0.5 - R_y, (R_x - 0.5)\,\mathrm{sgn}(\cos\omega_\phi) + \sin\omega_\phi, R_a - d_a]^T \in \mathbb{R}^3. \quad (12)$$

3.2.9 PST

Pedestal/Elevator Shot with Target (PST) can be described as the following set of requirements:

- The UAV should retain constant speed v_v along the vertical axis. This is an external value that implicitly determines shot duration.
- The camera axis yaw angle should remain fixed to $\phi_C = 0$.
- Target 2D ROI area R_a should remain approximately fixed to d_a. d_a is an external value that implicitly determines the desired FST.

- Target ROI center $[R_x, R_y]^T$ should always remain at the video frame center/principal point.

The following controllers must be constructed, based on the above requirements:

- Instant camera pitch rotation angle $\Delta\theta_C$.
- Instant UAV speed along the horizontal axis Δv_h.
- The camera focal length f.

Therefore, the PST error signal can be written as follows:

$$\mathbf{e}(t) = [0.5 - R_y, R_x - 0.5, R_a - d_a]^T \in \mathbb{R}^3. \quad (13)$$

3.2.10 CONLTS

Constrained Lateral Tracking Shot (CONLTS) can be described as the following set of requirements:

- The camera axis pitch angle should remain fixed to $\theta_C > 0$. This is an external value that implicitly determines the UAV altitude relative to the target.
- The camera axis yaw angle should remain fixed to $\phi_C = 0$.
- Target 2D ROI area R_a should remain approximately fixed to d_a. d_a is an external value that implicitly determines the desired FST.
- Target ROI center $[R_x, R_y]^T$ should always remain at the video frame center/principal point.

The following controllers must be constructed, based on the above requirements:

- Instant UAV speed along the horizontal axis Δv_h.
- Instant UAV speed along the vertical axis Δv_v.
- The camera focal length f.

Therefore, the CONLTS error signal can be written as follows:

$$\mathbf{e}(t) = [R_x - 0.5, R_y - 0.5, R_a - d_a]^T \in \mathbb{R}^3. \quad (14)$$

4 Evaluation

4.1 Evaluation Setup

AirSim, i.e., a photorealistic UAV simulation environment, was employed and extended in order to evaluate the proposed vision-driven PID controllers. AirSim [55] is built on top of the advanced Unreal 4 (UE4) real-time 3D graphics/physics engine. It allows programmatic interaction with the simulated UAVs via Remote Procedural Call (RPC)-based communication.

In the conducted evaluation sessions, a cycling scenario on a mountainous environment was simulated. A target cyclist was set to traverse a predefined route, which is of course unknown to the filming UAV. The latter was set to follow and shoot the cyclist while executing the desired CMTs, without having any prior knowledge of the environment, of the bicycle's track or of the cyclist's motion patterns. This is a challenging scenario, since the target moves along a non-linear route with several turns and curves. During the simulation, the cyclist maintained an approximately (not fully) constant speed. This scalar speed was also unknown to the UAV, except for the cases of ORBIT, ASCENT and DESCENT where it must be known a priori, as described in Subsection 3.2. It must be noted that several of the implemented UAV CMTs are ideally defined for linearly moving targets [35], due to aesthetic considerations. An example video frame captured by the simulated UAV while executing an LTS CMT is depicted in Figure 2.

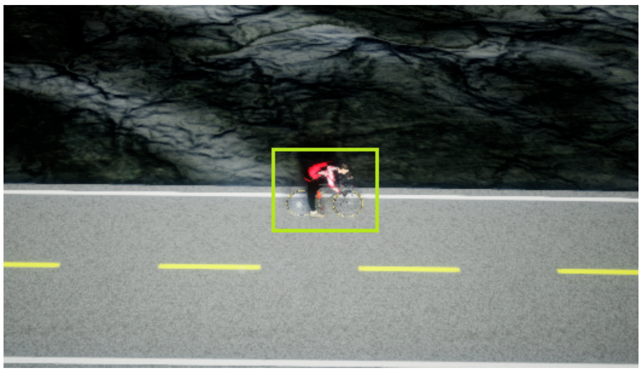


Fig. 2: An example video frame of executing an LTS on a cyclist, detected via YOLOv2, inside the AirSim simulator.

The overall architecture, depicted in Figure 3, consisted of: a) an environment along with the AirSim server, b) a detector/tracker c) PID controllers d) multiple AirSim clients for extracting and sending back information to the server (e.g. extracting images from the UAV's camera, sending UAV/gimbal control commands to the server, etc.). The communication between the modules is accomplished through ROS, excluding the RPC-based interaction between the AirSim server and clients.

The integrated detector/tracker module, based on [43] [42] and updated in [49], consisted of the YOLOv2 Detector [52] and the SiamRPN tracker [30]. The detector's input resolution was set to 300x300. The SiamRPN/YOLOv2 tracker/detector was pretrained on the OTB dataset [59] /real-world image dataset reported in [42], respectively; no detector/tracker fine-tuning for the AirSim evaluation environment was performed. Overall, this is more of a typical visual detection/tracking setup, rather than an absolute state-of-the-art, since the goal was to evaluate the proposed system performance in combination with standard, off-the-shelf auxiliary software. Therefore, in order to discriminate be-

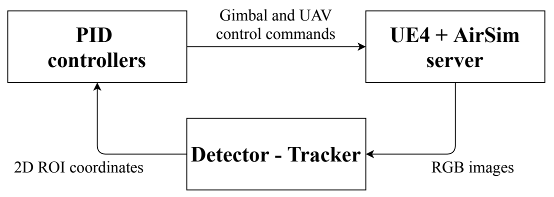


Fig. 3: The evaluation setup architecture. For simplicity, multiple AirSim clients are omitted.

tween performance drops due to controller inadequacy and due to detector/tracker failures, two sets of evaluation scenarios were conducted and are reported: one with ground-truth target ROIs (*GT-Vision*) and one with detector/tracker-derived target ROIs (*DT-Vision*). Implementation-wise, PID coefficients $\mathbf{K}_p$, $\mathbf{K}_i$, $\mathbf{K}_d$ were empirically discovered per controller, via manual tuning, and used for both of these vision-driven control modes. The complete footage captured through the simulator for all CMTs, as well as the optimal PID coefficients used for the experiments, are publicly available[1].

4.2 Evaluation Results

Two evaluation protocols were followed: an objective one and a subjective one. The goal of the objective evaluation was to assess whether the UAV trajectory relative to the target being filmed under vision-driven control, while capturing a specific shot, was correlated to the corresponding trajectories under 3D state-based control. The latter control mode (*3D*) was simply realized by implementing the reference CMT description equations from [35], which assume full UAV/target 3D state knowledge at all times and output UAV trajectory waypoints for executing the desired CMT at a fixed temporal rate of F Hz (i.e., the next desired UAV 3D position is computed F times per second). Each waypoint was fed to the simulated Pixhawk/PX4 autopilot immediately after its computation.

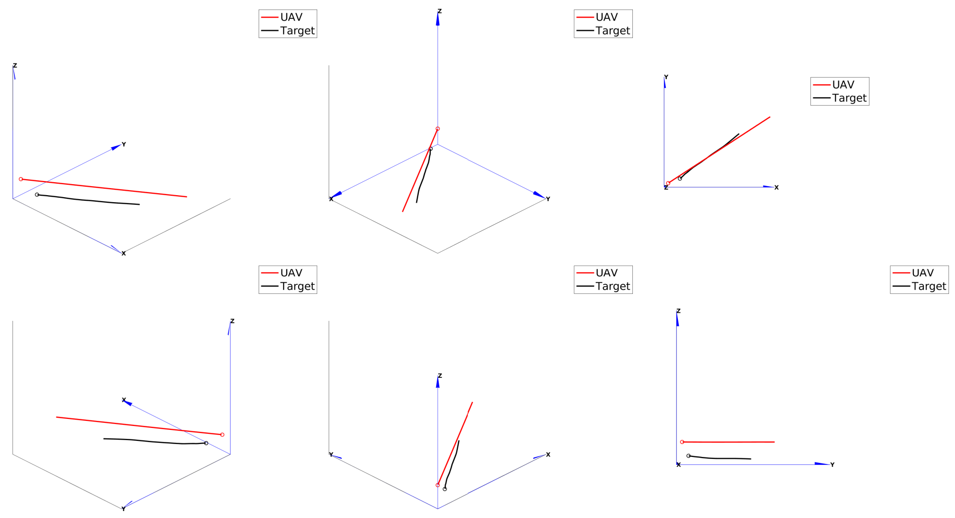
Tables 4, 5 and 6 depict the objective evaluation results. The normalized cross-correlation of target-relative 3D UAV positions over a full CMT trajectory is shown across three pairs: between the GT-Vision and the reference 3D control mode, between the DT-Vision and the reference 3D control mode, as well as between the two vision-driven control modes themselves (GT-Vision and DT-Vision). The results are depicted separately for each of the three axes of the Cartesian world coordinate system employed by AirSim and for each of the supported CMTs, while the mean across all axes is also depicted per CMT. Note that normalized cross-correlation lies in the interval $[-1.0, 1.0]$, with 1.0 indicating the strongest possible dependence between signals and 0.0 implying a complete lack of dependence.

Evidently, in the majority of cases, both GT-Vision and DT-Vision control modes achieve very high trajectory cross-correlation with the reference 3D mode in all three axes of the Cartesian world coordinate system employed by AirSim. DT-Vision correlation is slightly lower due to detector/tracker failures, which is to be expected, but the minor deviation from the performance of GT-Vision suggests that the proposed method can be readily employed in actual systems with off-the-shelf, pretrained detectors/trackers. Notably, the 3D UAV trajectories obtained in DT-Vision mode are very highly correlated with the corresponding trajectories derived by exploiting ground-truth target ROIs in GT-Vision mode, implying that the proposed controllers are relatively robust to 2D visual detection/tracking failures.

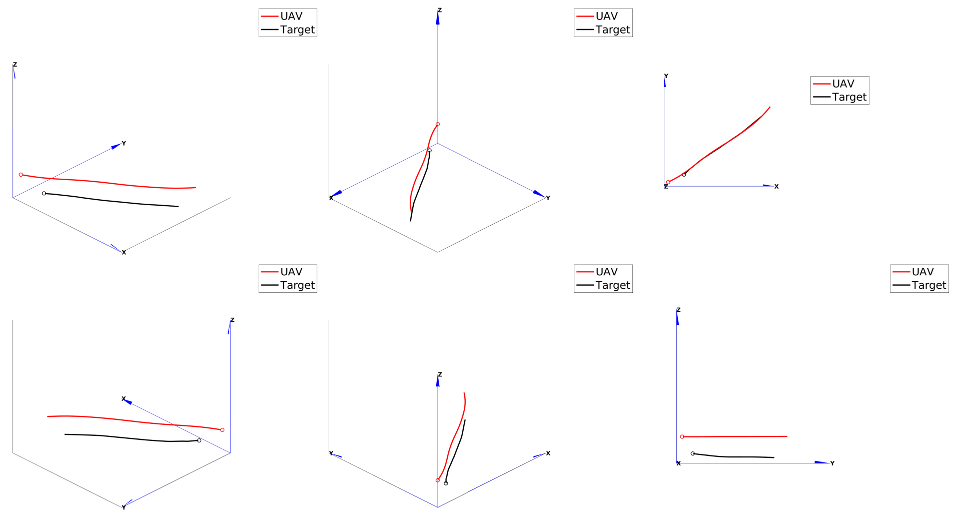
Overall, the lowest/worst normalized cross-correlation (NCC) between the vision-driven and the reference 3D control modes is achieved by the FLYOVER controllers, rendering it the only problematic CMT. Although its mean NCC with the 3D control mode UAV trajectory is $0.8043/0.6700$ for GT-Vision/DT-Vision, the corresponding per-axis NCC drops as low as 0.2400 for the X-axis in DT-Vision mode (still significantly higher than 0, but relatively low). However, by contrasting the FLYOVER trajectories obtained by all three control modes, we can see that the CMT is indeed performed correctly in all cases. This is depicted in Figure 4, where each of the three plots is shown from six different view angles and, in all cases, the small red/black circle denotes the start of the UAV/target trajectory. The relatively low NCC scores are evidently due to the “jittery” UAV trajectories achieved in vision-driven control modes, instead of the purely linear one in reference 3D control mode. Although this seems to simply be an issue fixable by proper fine-tuning of the PID coefficients in the FLYOVER controllers, we were unable to come up with a better set of parameters, despite the large expended effort. In general, PID control is known to be sensitive to coefficient selection.

The goal of subjective evaluation was to let human judges deduce in a systematic manner whether the visual result of the three control modes (GT-Vision, DT-Vision and 3D), for each supported CMT: a) is *representative* of the ideal CMT from a cinematography/aesthetics perspective, and b) is visually *enjoyable*. In all cases, a single-stimulus Absolute Category Rating (ACR) methodology was used [20], while 10 subjects were employed: 8 naive and 2 experts. In each trial, a participant was shown a test sequence per CMT for each control mode (without being informed of which mode it was drawn from) and had unlimited time to grade this sequence in terms of representativeness and enjoyability. The discrete scale used for both metrics is the following one: 5=Excellent, 4=Good, 3=Fair, 2=Poor and 1=Bad. A training session took place for each subject before formal eval-

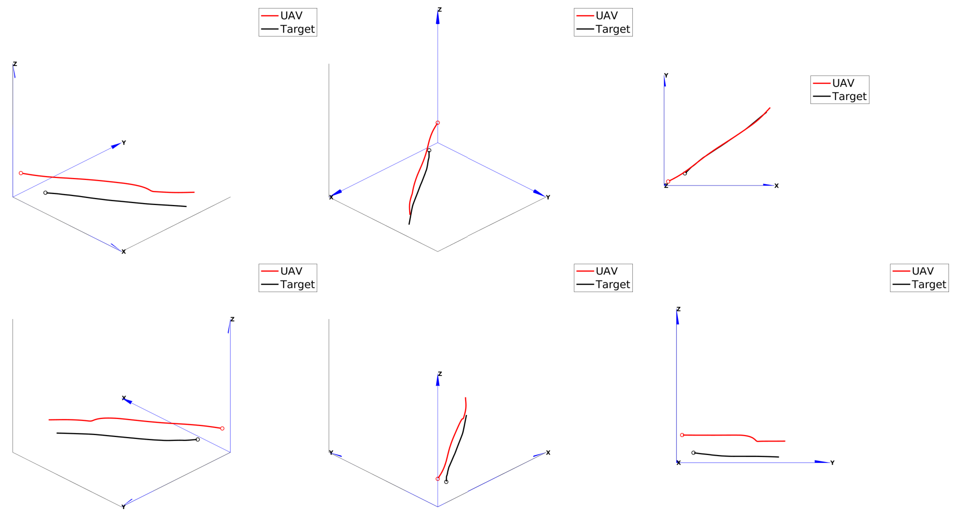
[1] `https://drive.google.com/drive/folders/1RhxG0PbWIzrNHk3_1YQGaKU6CNz5LxpN`



(a)



(b)



(c)

Fig. 4: FLYOVER 3D trajectories in: a) reference 3D control mode, b) GT-Vision mode, and c) DT-Vision mode.

CMT	X-axis	Y-axis	Z-axis	Mean
ORBIT	0.8873	0.8900	**0.9630**	0.9134
CHASE	0.9706	0.9991	0.9994	0.9897
VTS	0.5976	1.0000	1.0000	0.8658
LTS	0.9165	0.9991	0.9998	0.9718
FLYBY	0.9272	0.9699	0.9993	0.9655
FLYOVER	**0.5491**	**0.8641**	0.9997	**0.8043**
DESCENT	0.9990	0.9996	0.9998	0.9995
ASCENT	0.9999	1.0000	1.0000	1.0000
PST	0.9999	1.0000	1.0000	1.0000
CONLTS	0.9724	0.8871	0.9998	0.9531

Table 4: Normalized cross-correlation between the GT-Vision and the reference 3D control modes. The two signals are target-relative 3D UAV positions over time, during the execution of each CMT. All values lie in $[-1.0, 1.0]$, where higher is better. The lowest value per column is highlighted in bold.

CMT	X-axis	Y-axis	Z-axis	Mean
ORBIT	0.8806	0.8943	**0.9615**	0.9121
CHASE	0.9635	0.9985	0.9989	0.9870
VTS	0.7809	1.0000	1.0000	0.9270
LTS	0.8669	0.9985	0.9997	0.9551
FLYBY	0.9277	0.9702	0.9992	0.9657
FLYOVER	**0.2400**	**0.7706**	0.9995	**0.6700**
DESCENT	0.9975	0.9988	0.9998	0.9987
ASCENT	1.0000	1.0000	1.0000	1.0000
PST	0.9999	0.9999	1.0000	1.0000
CONLTS	0.9783	0.9487	0.9998	0.9756

Table 5: Normalized cross-correlation between the DT-Vision and the reference 3D control modes. The two signals are target-relative 3D UAV positions over time, during the execution of each CMT. All values lie in $[-1.0, 1.0]$, where higher is better. The lowest value per column is highlighted in bold.

uation, consisting of showing him/her indicative sequences per CMT (extracted from real UAV video footage).

Tables 8 and 7 depict the results of the subjective evaluation. As expected, in terms of representativeness, most CMTs executed in 3D control mode achieved slightly higher scores compared to GT-Vision and DT-Vision modes. However, certain CMTs such as ORBIT, ASCENT, DESCENT and PST, are relatively on par for all three control modes. Enjoyability scores were also typically a little lower for GT-Vision/DT-Vision modes in comparison to 3D control mode results, but the differences range from small to negligible. Notably, 4 out of 10 CMTs appear more enjoyable in GT/DT-Vision modes than in 3D control mode. This includes the FLYOVER CMT, indicating that the "jittery" trajectories depicted in Figure 4 are not non-linear/non-smooth enough to cause discomfort to the viewer. In general, the best-performing CMTs in GT/DT-Vision modes are CHASE, FLYOVER, DESCENT, ASCENT, PST and CONLTS.

CMT	X-axis	Y-axis	Z-axis	Mean
ORBIT	0.9964	0.9991	**0.9996**	0.9984
CHASE	0.9978	0.9995	0.9998	0.9990
VTS	**0.7077**	1.0000	1.0000	**0.9026**
LTS	0.8717	0.9984	0.9997	0.9566
FLYBY	0.9997	0.9999	1.0000	0.9999
FLYOVER	0.9413	**0.9862**	1.0000	0.9758
DESCENT	0.9996	0.9998	1.0000	0.9998
ASCENT	1.0000	1.0000	1.0000	1.0000
PST	1.0000	1.0000	1.0000	1.0000
CONLTS	0.9995	0.9872	1.0000	0.9956

Table 6: Normalized cross-correlation between the GT-Vision and the DT-Vision control modes. The two signals are target-relative 3D UAV positions over time, during the execution of each CMT. All values lie in $[-1.0, 1.0]$, where higher is better. The lowest value per column is highlighted in bold.

CMT	3D	GT-Vision	DT-Vision
ORBIT	**4.7**	3.6	3.5
CHASE	**4.9**	4.1	3.7
VTS	**5.0**	4.0	4.2
LTS	**4.9**	3.6	3.8
FLYBY	**4.6**	3.4	3.4
FLYOVER	3.7	**4.0**	3.7
DESCENT	**5.0**	**5.0**	**5.0**
ASCENT	**4.9**	4.6	4.6
PST	**5.0**	4.7	4.6
CONLTS	**4.8**	4.3	3.9

Table 7: Mean subjective evaluation scores for the representativeness metric. All values are in $[1.0, 5.0]$, where higher is better. The highest value per row is highlighted in bold.

CMT	3D	GT-Vision	DT-Vision
ORBIT	**4.5**	4.0	3.9
CHASE	4.4	**4.6**	4.2
VTS	**4.9**	3.5	3.7
LTS	**4.9**	3.5	3.4
FLYBY	**4.3**	3.8	3.8
FLYOVER	3.4	**4.2**	3.6
DESCENT	4.6	**4.8**	4.6
ASCENT	**5.0**	4.6	4.6
PST	**5.0**	4.9	**5.0**
CONLTS	4.5	4.0	**4.7**

Table 8: Mean subjective evaluation scores for the enjoyability metric. All values are in $[1.0, 5.0]$, where higher is better. The highest value per row is highlighted in bold.

Further evaluation was conducted using an alternative combination of more recent and improved neural object detectors/trackers: YOLOv5s [21] and SiamAPN++ [9], instead of YOLOv2/SiamRPN. Being fast and lightweight, they too are suitable for embedded AI applications. SiamAPN++ was pretrained on the UAV20L dataset. YOLOv5s was pretrained on the COCO dataset and finetuned on two cycling detection/tracking datasets [2] [42]. The achieved normalized cross-correlation between the DT-Vision control mode, using YOLOv5s/SiamAPN++ as detector/tracker, and the reference 3D control mode, averaged across all CMTs, was 0.943. This is a very minimal gain of +0.4%, compared to the one achieved using YOLOv2/SiamRPN in the vision subsystem. The most notable gains were attained in FlYOVER and LTS, achieving +4.1% and +1.1% respectively.

4.3 Discussion

Overall, subjective evaluation results for both metrics show that CMTs executed in GT/DT-Vision modes are comparably pleasing and visually similar to those in 3D control mode. The entire evaluation process indicates that controllers designed according to the proposed method are unexpectedly robust to a degree of 2D visual detection/tracking failure, but also confirms their unsurprising sensitivity to PID coefficients selection.

Assuming properly tuned coefficients, the proposed method indeed leads to autonomous execution of desired UAV CMTs without 3D state information or estimation, relying solely on 2D visual input. Certain UAV/camera parameters that are required to be known, as detailed in Section 3, can in fact be always kept internally up-to-date trivially, using only the on-board IMU(s). Of course, the target must first be visibly located within the camera's field of view, during shot initialization at the first time instance, so that its 2D image can be localized by a real-time 2D visual detector/tracker. Therefore, the method is most appropriate for scenarios where rough GPS position signals are available (so that the camera/drone can be initially rotated in order to approximately face the target), but they are inaccurate, unstable and/or intermittent. In fully GPS-denied environments the gimbal/drone may be manually rotated during shot initialization (at the first video frame/time instance) so that the target lies within the camera's field-of-view; from then on, the method may proceed fully autonomously. Finally, if the goal is to execute a CMT relative to the first target of a specific type that happens to become visible, neither rough GPS positions nor manual gimbal/drone rotation are required during shot initialization.

An important conclusion drawn from the evaluation process is the relatively high method robustness to object detection/tracking failures. The captured CMT in DT-Vision mode is approximately correct, even in the presence of object detection/tracking errors/instabilities concerning target 2D ROI localization across the video sequence. Since highly robust, state-of-the-art, off-the-shelf deep neural object detectors/trackers are currently available, that can be executed in real-time on embedded AI compute boards (e.g., nVidia Jetson), the proposed method seems rather resilient to typical levels of ROI localization noise. This robustness is also implicitly confirmed by the very minimal gains in CMT execution accuracy when employing a more recent object detector/tracker combination: despite obtaining (on average) a more precise target 2D ROI at each video frame, the proposed method's ability to autonomously execute UAV CMTs remains almost identical for the vast majority of implemented shot types.

As a side-note, despite the high importance of CMTs for UAV cinematography applications, other cases where CMTs may prove useful can be imagined. For instance, if a landing site is visually detected on-frame, a typical relevant CMT for landing would be to fly in parallel to the local tangent plane towards the detected site and, once the UAV lies exactly above it, start flying down. A CMT could be easily described for this process according to the proposed method, resulting automatically in vision-driven PID controllers that autonomously implement it.

Finally, given that tuning and evaluation of the proposed method took place in a simulated environment, the issue of transfer to real-world UAVs inevitably arises. This is the so-called "reality gap", commonly encountered in the robotics literature. In this case, there are two complementary aspects to the issue: i) how well the deep neural object detector/tracker subsystem would perform in the real world, and ii) how well would the controller parameters that have been manually tuned in simulation (e.g., the PID coefficients per CMT) perform in the real world.

In our opinion the first aspect is not that important, given the significant robustness against detection/tracking errors empirically demonstrated by the implemented controllers and the constant improvement of deep neural architectures able to run in real-time on embedded AI hardware. In fact, if an object detector/tracker has been properly trained on real-world image datasets, it tends to perform well in both real-world test images and within a photorealistic simulation environment (it is the opposite scenario that does not hold). Therefore, in our case, there is no significant issue: the employed vision subsystem was pretrained on real-world datasets and is known to be running successfully in real-time on embedded AI platforms.

However, the second aspect is more problematic: given the sensitivity the implemented controllers empirically demonstrated to PID coefficient values, a drop in performance is to be expected when the reality gap is crossed. The trivial solution would be to increase the noise in the execution of simulated actions [3] and then retune the controller parameters from scratch. Most likely, this would result in lower achievable accuracy which, however, would remain almost constant after transfer to a real-world UAV. A better, much

[2] *Benchmark_RAI* and *Annotations_Bicycles_Raw* datasets were downloaded from `https://aiia.csd.auth.gr/open-multidrone-datasets/`

[3] A default level of noise is already inserted by the simulator.

more involved solution would be to replace the simple PID controllers of the proposed method with self-tuning adaptive PID controllers; these are able to periodically retune their parameters on their own, in response to changes in their operational environment [3]. Transfer from simulation to a real UAV would certainly qualify as such a change, thus minimizing the observed drop in performance. This is, in fact, a very interesting future research direction.

5 Conclusions

Professional filming with camera-equipped Unmanned Aerial Vehicles (UAVs, or drones) depends on specifying and executing a sequence of desired Camera Motion Types (CMTs), typically with regard to a filming target/subject. Automating this process with autonomous, robotic UAVs typically requires knowledge of the target's 3D state at all time instances (e.g., through on-target GPS). However, this is cumbersome, or even impossible when filming non-scripted scenes, and/or when flying in GPS-denied outdoor environments. As an alternative, this paper proposes a novel method for purely vision-driven PID control of UAVs, in order to autonomously execute the desired target-tracking CMTs, based solely on what the drone-mounted camera "sees" in 2D pixel coordinates at each time instance, without the need for knowing 3D information. To the best of our knowledge, no previous method of vision-driven UAV control for autonomous CMT execution has been systematically developed in the intelligent shooting community. Only rudimentary, non-generic and unpublished/unknown algorithms of this general category, each one specifically designed for a given CMT, seem to be implemented in commercial, proprietary UAVs equipped with autonomous functionalities.

As a concrete example of the proposed method, a set of industry-standard, formalized UAV cinematography CMTs have been reformulated in the form of requirements that interrelate 2D visual information, 3D UAV trajectory and camera orientation. Following the proposed method, this has facilitated the implementation of a corresponding set of vision-driven PID controllers per CMT, that exploit the above requirements to form error signals driving continuous adjustments to instant UAV motion parameters. The end result is autonomous execution of each supported CMT. Depth maps, target 3D models or extrinsic/intrinsic camera parameters are not needed. The only required input is the target's bounding box in 2D pixel coordinates at each video frame, as detected by an independent, real-time, off-the-shelf deep neural 2D visual detector/tracker. The proposed method is significantly more robust and reliable than target 3D state estimation from visual data and camera parameters. The main novelty lies in the fact that neither UAV nor target 3D state inputs are required to be known or estimated, thus making this method suitable for filming with inaccurate or intermittent GPS signals. Objective and subjective evaluation in a photorealistic UAV simulator indicates that the proposed PID controllers are powerful enough to permit successful autonomous execution of the supported CMTs, potentially opening up new avenues in autonomous UAV cinematography. Additionally, the method is generic enough to be applicable to non-cinematography settings as well, as long as a specific UAV motion trajectory relative to a visible target/subject is desired.

Given the robustness the proposed method empirically demonstrated to failures of the neural detector/tracker, as well as the continuing improvements of embedded Deep Neural Networks (DNNs) in computer vision, it seems inevitable that similar algorithms will become common place in the near future. Indeed, cognitive autonomy is increased with each passing generation of commercial multicopters for videography/cinematography, leading us to assume that near-future professional UAVs will be able to film almost entirely on their own, if provided a valid cinematography plan by the user. On the other hand, this does not make the human pilot obsolete: DNNs still struggle with adaptation to dynamically changing environments, while controller-specific parameters (such as the per-CMT PID coefficients in the proposed method) may also fail in the presence of scene variations over time. As a result, the current near-future trend is to treat such autonomous functionalities as a valuable tool in the hands of the human operator, rather than a substitute of them.

Funding and/or Conflicts of interests/Competing interests

The research leading to these results has received funding from the European Union's European Union Horizon 2020 research and innovation programme under grant agreement No 731667 (MULTIDRONE). This publication reflects only the author's views. The European Union is not liable for any use that may be made of the information contained therein.

The authors have no competing interests to declare that are relevant to the content of this article.

Data availability

The datasets generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

References

1. Alcantara, A., Capitan, J., Cunha, R., Ollero, A.: Optimal trajectory planning for cinematography with multiple Unmanned Aerial Vehicles. Robotics and Autonomous Systems **140**, 103778 (2021)

2. Alcantara, A., Capitan, J., Torres-Gonzalez, A., Cunha, R., Ollero, A.: Autonomous execution of cinematographic shots with multiple drones. IEEE Access pp. 201300–201316 (2020)
3. Alexandrov, A.G., Palenov, M.V.: Adaptive PID controllers: State of the art and development prospects. Automation and remote control **75**(2), 188–199 (2014)
4. Åström, K.J., Hägglund, T., Astrom, K.J.: Advanced PID control, vol. 461. ISA-The Instrumentation, Systems, and Automation Society Research Triangle (2006)
5. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014)
6. Bhattacharya, S., Mehran, R., Sukthankar, R., Shah, M.: Classification of cinematographic shots using lie algebra and its application to complex event recognition. IEEE Transactions on Multimedia **16**(3), 686–696 (2014)
7. Bonatti, R., Ho, C., Wang, W., Choudhury, S., Scherer, S.: Towards a robust aerial cinematography platform: Localizing and tracking moving targets in unstructured environments. arXiv preprint arXiv:1904.02319 (2019)
8. Bradski, G., Kaehler, A., Pisarevsky, V.: Learning-based computer vision with Intel's open source computer vision library. Intel Technology Journal **9**(2), 119–130 (2005)
9. Cao, Z., Fu, C., Ye, J., Li, B., Li, Y.: SiamAPN++: Siamese attentional aggregation network for real-time UAV tracking. In: Proceedings of the International Conference on Intelligent Robots and Systems (IROS) (2021)
10. Caraballo, L.E., Montes-Romero, A., Diaz-Bañez, J.M., Capitan, J., Torres-Gonzalez, A., Ollero, A.: Autonomous planning for multiple aerial cinematographers. In: Proceedings of the International Conference on Intelligent Robots and Systems (IROS) (2020)
11. Carrio, A., Sampedro, C., Rodriguez-Ramos, A., Campoy, P.: A review of deep learning methods and applications for unmanned aerial vehicles. Journal of Sensors **2017** (2017)
12. Chen, C., Ling, Q.: Adaptive convolution for object detection. IEEE Transactions on Multimedia **21**(12), 3205–3217 (2019)
13. Devo, A., Dionigi, A., Costante, G.: Enhancing continuous control of mobile robots for end-to-end visual active tracking. Robotics and Autonomous Systems **142**, 103799 (2021)
14. Fourati, H., Belkhiat, D.: Multisensor Attitude Estimation: Fundamental Concepts and Applications. CRC Press LLC (2016)
15. Gao, X.S., Hou, X.R., Tang, J., Cheng, H.F.: Complete solution classification for the perspective-three-point problem. IEEE Transactions on Pattern Analysis and Machine Intelligence **25**(8), 930–943 (2003)
16. Grewal, M.S., Weill, L.R., Andrews, A.P.: Global Positioning Systems, inertial navigation, and integration. John Wiley & Sons (2007)
17. Gschwindt, M., Camci, E., Bonatti, R., Wang, W., Kayacan, E., Scherer, S.: Can a robot become a movie director? learning artistic principles for aerial cinematography. arXiv preprint arXiv:1904.02579 (2019)
18. Huang, C., Lin, C.E., Yang, Z., Kong, Y., Chen, P., Yang, X., Cheng, K.T.: Learning to film from professional human motion videos. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2019)
19. Huang, C., Yang, Z., Kong, Y., Chen, P., Yang, X., Cheng, K.T.T.: Learning to capture a film-look video with a camera drone. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) (2019)
20. Huynh-Thu, Q., Garcia, M.N., Speranza, F., Corriveau, P., Raake, A.: Study of rating scales for subjective quality assessment of high-definition video. IEEE Transactions on Broadcasting **57**(1), 1–14 (2010)
21. Jocher, G.e.a.: ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation (2022). URL `https://doi.org/10.5281/zenodo.7347926`
22. Joubert, N., Goldman, D.B., Berthouzoz, F., Roberts, M., Landay, J.A., Hanrahan, P.: Towards a drone cinematographer: Guiding quadrotor cameras using visual composition principles. arXiv preprint arXiv:1610.01691 (2016)
23. Kakaletsis, E., Symeonidis, C., Tzelepi, M., Mademlis, I., Tefas, A., Nikolaidis, N., Pitas, I.: Computer vision for autonomous UAV flight safety: An overview and a vision-based safe landing pipeline example. ACM Computing Surveys (CSUR) **54**(9), 1–37 (2021)
24. Karakostas, I., Mademlis, I., Nikolaidis, N., Pitas, I.: UAV cinematography constraints imposed by visual target tracking. In: Proceedings of the IEEE International Conference on Image Processing (ICIP) (2018)
25. Karakostas, I., Mademlis, I., Nikolaidis, N., Pitas, I.: Shot type feasibility in autonomous UAV cinematography. In: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (2019)
26. Karakostas, I., Mademlis, I., Nikolaidis, N., Pitas, I.: Shot type constraints in UAV cinematography for autonomous target tracking. Information Sciences **506**, 273–294 (2020)
27. Kelchtermans, K., Tuytelaars, T.: How hard is it to cross the room? Training (Recurrent) Neural Networks to steer a UAV. arXiv preprint arXiv:1702.07600 (2017)
28. Kim, D., Chen, T.: Deep neural network for real-time autonomous indoor navigation. arXiv preprint arXiv:1511.04668 (2015)
29. Kuang, Q., Jin, X., Zhao, Q., Zhou, B.: Deep multimodality learning for UAV video aesthetic quality assessment. IEEE Transactions on Multimedia (2019)
30. Li, B., Yan, J., Wu, W., Zhu, Z., Hu, X.: High performance visual tracking with siamese Region Proposal Network. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
31. Li, G., Mueller, M., Casser, V., Smith, N., Michels, D.L., Ghanem, B.: OIL: Observational Imitation Learning. arXiv preprint arXiv:1803.01129 (2018)
32. Mademlis, I., Mygdalis, V., Nikolaidis, N., Montagnuolo, M., Negro, F., Messina, A., Pitas, I.: High-level multiple-UAV cinematography tools for covering outdoor events. IEEE Transactions on Broadcasting **65**(3), 627–635 (2019)
33. Mademlis, I., Mygdalis, V., Nikolaidis, N., Pitas, I.: Challenges in autonomous UAV cinematography: an overview. In: Proceedings of the IEEE International Conference on Multimedia and Expo (ICME) (2018)
34. Mademlis, I., Nikolaidis, N., Tefas, A., Pitas, I., Wagner, T., Messina, A.: Autonomous unmanned aerial vehicles filming in dynamic unstructured outdoor environments. IEEE Signal Processing Magazine **36**, 147–153 (2018)
35. Mademlis, I., Nikolaidis, N., Tefas, A., Pitas, I., Wagner, T., Messina, A.: Autonomous UAV cinematography: A tutorial and a formalized shot type taxonomy. ACM Computing Surveys **52**(5), 105 (2019)
36. Mademlis, I., Torres-González, A., Capitán, J., Montagnuolo, M., Messina, A., Negro, F., Le Barz, C., Gonçalves, T., Cunha, R., Guerreiro, B., et al.: A multiple-UAV architecture for autonomous media production. Multimedia Tools and Applications pp. 1–30 (2022)
37. Meier, L., Tanskanen, P., Fraundorfer, F., Pollefeys, M.: Pixhawk: A system for autonomous flight using onboard computer vision. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) (2011)
38. Mur-Artal, R., Tardós, J.D.: Visual-inertial monocular SLAM with map reuse. IEEE Robotics and Automation Letters **2**(2), 796–803 (2017)
39. Nägeli, T., Alonso-Mora, J., Domahidi, A., Rus, D., Hilliges, O.: Real-time motion planning for aerial videography with dynamic obstacle avoidance and viewpoint optimization. IEEE Robotics and Automation Letters **2**(3), 1696–1703 (2017)

40. Nägeli, T., Meier, L., Domahidi, A., Alonso-Mora, J., Hilliges, O.: Real-time planning for automated multi-view drone cinematography. ACM Transactions on Graphics **36**(4), 132:1–132:10 (2017)
41. Naseer, T., Sturm, J., Cremers, D.: Followme: Person following and gesture recognition with a quadrocopter. In: Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS) (2013)
42. Nousi, P., Mademlis, I., Karakostas, I., Tefas, A., Pitas, I.: Embedded UAV real-time visual object detection and tracking. In: Proceedings of the IEEE International Conference on Real-time Computing and Robotics (RCAR) (2019)
43. Nousi, P., Patsiouras, E., Tefas, A., Pitas, I.: Convolutional Neural Networks for visual information analysis with limited computing resources. In: Proceedings of the IEEE International Conference on Image Processing (ICIP) (2018)
44. Papaioannidis, C., Mademlis, I., Pitas, I.: Autonomous UAV safety by visual human crowd detection using multi-task deep neural networks. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) (2021)
45. Papaioannidis, C., Mademlis, I., Pitas, I.: Fast CNN-based single-person 2D human pose estimation for autonomous systems. IEEE Transactions on Circuits and Systems for Video Technology (2022)
46. Passalis, N., Tefas, A.: Deep reinforcement learning for controlling frontal person close-up shooting. Neurocomputing **335**, 37–47 (2019)
47. Passalis, N., Tefas, A., Pitas, I.: Efficient camera control using 2D visual information for unmanned aerial vehicle-based cinematography. In: Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS) (2018)
48. Patrona, F., Mademlis, I., Tefas, A., Pitas, I.: Computational UAV cinematography for intelligent shooting based on semantic visual analysis. In: Proceedings of the IEEE International Conference on Image Processing (ICIP) (2019)
49. Patrona, F., Nousi, P., Mademlis, I., Tefas, A., Pitas, I.: Visual object detection for autonomous UAV cinematography. In: Proceedings of the Northern Lights Deep Learning Workshop (2020)
50. Piao, J., Kim, S.: Real-time visual–inertial SLAM based on adaptive keyframe selection for mobile AR applications. IEEE Transactions on Multimedia **21**(11), 2827–2836 (2019)
51. Ranjan, R., Patel, V.M., Chellappa, R.: Hyperface: A deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence **41**(1), 121–135 (2017)
52. Redmon, J., Farhadi, A.: Yolo9000: Better, faster, stronger. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
53. Sadeghi, F., Levine, S.: (CAD)2RL: Real single-image flight without a single real image. arXiv preprint arXiv:1611.04201 (2016)
54. Serra, P., Cunha, R., Hamel, T., Cabecinhas, D., Silvestre, C.: Landing of a quadrotor on a moving target using dynamic image-based visual servo control. IEEE Transactions on Robotics **32**(6), 1524–1535 (2016)
55. Shah, S., Dey, D., Lovett, C., Kapoor, A.: AirSim: High-fidelity visual and physical simulation for autonomous vehicles. In: Proceedings of the Field and Service Robotics Conference (2017)
56. Symeonidis, C., Mademlis, I., Nikolaidis, N., Pitas, I.: Improving neural Non-Maximum Suppression for object detection by exploiting interest-point detectors. In: Proceedings of the IEEE International Workshop on Machine Learning for Signal Processing (MLSP) (2019)
57. Teuliere, C., Eck, L., Marchand, E.: Chasing a moving target from a flying UAV. In: Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS) (2011)
58. Teuliere, C., Marchand, E.: A dense and direct approach to visual servoing using depth maps. IEEE Transactions on Robotics **30**(5), 1242–1249 (2014)
59. Wu, Y., Lim, J., Yang, M.: Online object tracking: A benchmark. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2013)
60. Zhong, F., Sun, P., Luo, W., Yan, T., Wang, Y.: AD-VAT: An asymmetric dueling mechanism for learning visual active tracking. In: Proceedings of the International Conference on Learning Representations (ICLR) (2018)