Free as a Bird: Event-based Dynamic Sense-and-Avoid for Ornithopter Robot Flight

J.P. Rodríguez-Gómez, R. Tapia, M.M. Guzmán, J.R. Martínez-de Dios and A. Ollero

***Abstract*— Autonomous flight of flapping-wing robots is a major challenge for robot perception. Most of the previous *sense-and-avoid* works have studied the problem of obstacle avoidance for flapping-wing robots considering only static obstacles. This paper presents a fully onboard dynamic *sense-and-avoid* scheme for large-scale ornithopters using event cameras. These sensors trigger pixel information due to changes of illumination in the scene such as those produced by dynamic objects. The method performs *event-by-event* processing in low-cost hardware such as those onboard small aerial vehicles. The proposed scheme detects obstacles and evaluates possible collisions with the robot body. The onboard controller actuates over the horizontal and vertical tail deflections to execute the avoidance maneuver. The scheme is validated in both indoor and outdoor scenarios using obstacles of different shapes and sizes. To the best of the authors' knowledge, this is the first event-based method for dynamic obstacle avoidance in a flapping-wing robot.**

***Index Terms*— event camera, ornithopter, flapping-wing robot, reactive sense-and-avoid.**

I. Introduction

Flapping-wing robots, also known as ornithopters, have recently attracted significant R&D interest. They can perform agile maneuvers [1] and combine flapping and gliding modes to reduce energy consumption [2]. Besides, flapping-wing robots are often made of soft materials making them less dangerous than multirotors in case of collision [3]. Flapping-wing flight describes novel perception challenges different from those in multirotor flight. First, ornithopters generate lift and thrust by flapping strokes, causing mechanical vibrations and wide abrupt movements that highly impact onboard perception [4]. Besides, they have strict payload and energy limitations, which strongly constrain the installation of sensors and additional hardware, involving strict limitations on the onboard processing capacity. In fact, most reported ornithopter perception and control methods, e.g., [5], [6], are executed offboard and use measurements from external sensors such as motion capture systems. We are interested in autonomous navigation of ornithopter robots, and particularly in avoidance of dynamic obstacles. While static obstacles can often be assumed within the map (and addressed through trajectory planning) or detected with additional sensors such as LiDAR, this work deals with the avoidance of unexpected dynamic obstacles in ornithopters, which require fast onboard detection and avoidance in contrast to the strict payload and resource constraints of these platforms. The perception scheme is based on event cameras. They are robust to motion blur and lighting conditions and have moderate weight and low energy consumption. Hence, they are suitable to deal with the flapping-wing flight perception challenges [4]. Besides, event cameras are suitable for dynamic obstacle detection by directly providing pixel information of moving objects in the scene. Additionally, efficient event-based processing techniques can provide estimates at very high rates. Many successful event-based perception techniques have been developed [7].

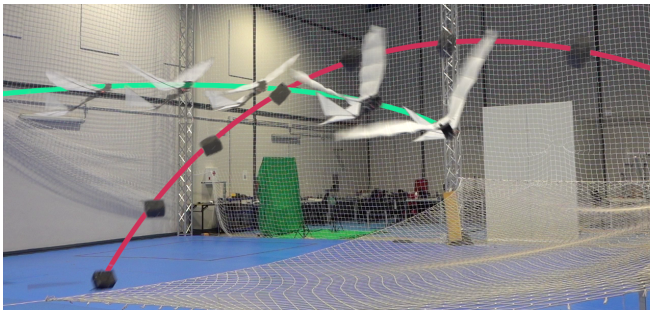

Fig. 1: Image sequence of the *E-Flap* robot performing an obstacle avoidance maneuver in an experiment.

This paper presents a dynamic obstacle *sense-and-avoid* method for ornithopter robots. By processing only one onboard event camera, the robot rapidly detects dynamic obstacles and modifies its trajectory to avoid them, exploiting the low latency of event cameras and the agility of ornithopters' flight. Dynamic obstacles are segmented using the spatio-temporal event information from objects that move with a different velocity than the background. An event optical flow method estimates its direction, and a reactive evasive maneuver strategy rapidly evaluates and prevents collisions. The method is implemented for online execution in resource-constrained hardware and is evaluated in the GRIFFIN *E-Flap* large-scale ornithopter [1], see Fig. 1, in indoor and outdoor experiments. To the best of the authors' knowledge, this is the first event-based obstacle avoidance method designed for and validated in flapping-wing robots.

The main contributions of the paper are: (1) an event-based dynamic object motion estimation method designed to perform in low-resource hardware; (2) a reactive obstacle avoidance method for large-scale ornithopters providing low latency onboard perception and control; and (3) experimental

The authors thank Jesus Tormo for his help with the robot electronics, and Angela Romero for her support on the validation experiments. This work was performed within GRIFFIN ERC Advanced Grant (Action 788247) (`https://griffin-erc-advanced-grant.eu`) funded by the European Research Council. Partial funding was obtained from European Commission H2020 AERIAL-CORE project (Grant H2020-2019-871479) and from Plan Estatal de Investigación Científica y Técnica y de Innovación of the Ministerio de Universidades del Gobierno de España (FPU19/04692).
The authors are with the GRVC Robotics Lab Sevilla. Universidad de Sevilla, Spain `email: {jrodriguezg, raultapia, mguzmang, jdedios, aollero}@us.es`

validation indoors and outdoors on a large-scale ornithopter.

This paper is organized as follows. Section II briefly summarizes the main works in the topics addressed in the paper. The general diagram of the event-based ornithopter dynamic obstacle avoidance scheme and its main components are described in Sections III and IV. Section V presents the experimental validation and robustness analyses. Section VI closes the paper and highlights the main future steps.

II. Related work

Reactive obstacle avoidance focuses on generating avoidance robot actions without relying on globally consistent map information. It can be categorized into *map-based* and *map-less* approaches [11]. The first builds a local map to compute obstacle-free trajectories [12], while the second, aims at detecting nearby obstacles and directly performs the avoidance action [8]. *Map-less* methods provide faster obstacle avoidance and are suitable for platforms with limited processing capacity. A work analysing the perception latency in a high-speed *sense-and-avoid* scenario with a quadrotor is presented in [13]. The work in [14] presents a *map-less* obstacle avoidance solution to avoid flying obstacles using a saliency-based reinforcement learning approach.

Recent advances in ornithopter development have led to the necessity of developing onboard perception methods capable of providing information for navigation, landing, and perching. Optical flow estimation onboard a Micro Aerial Vehicle (MAVs) flapping-wing robot is presented in [15]. The method sub-samples input images and uses a motion detection algorithm to compute the optical flow. Authors in [16] use the object appearance variation and optical flow to perform obstacle avoidance with a monocular camera. Obstacle detection and avoidance are computed in a ground station due to the weight limitation of the platform. A stereo-vision obstacle avoidance strategy for small-scale ornithopters is presented in [17]. The method computes sparse disparity maps from points with relatively high certainty for obstacle depth estimation. Obstacle avoidance is achieved through the droplet strategy defining the necessary obstacle-free area in front of the robot to guarantee safe avoidance maneuvers.

The advantages offered by event cameras have increased the research interest in computer vision and robotics communities [7]. Their temporal μs resolution and high dynamic range motivate their use for aerial robot perception. An event-based optical flow approach for autonomous MAV landing using a downwards orientated event camera is presented in [18]. The event-based line tracker in [19] provides fast and stable references for quadrotor visual servoing to perform bionspired landing trajectories. The drone racing dataset [20] including event data intends to encourage the development of perception methods for high-speed drone maneuvers. The work in [21] accumulates events to build *event images* in order to estimate the position and orientation of a quadrotor using a visual odometry method and closing the loop for autonomous flight subject to rotor failures. In [22], an auto-tuned event-based vision scheme performs intruder detection on board an autonomous quadrotor in surveillance missions.

Recently, some event-based methods for reactive obstacle avoidance for quadrotors have been presented. Work [8] describes an evasive method for dynamic obstacles using stereo event cameras on a quadrotor. The object trajectory is computed and propagated in time to predict collisions. An event-based dodging system for quadrotors is presented in [9]. It performs image deblurring, odometry estimation, and moving object segmentation using Deep Learning. Work [10] presents a dynamic obstacle avoidance method for quadrotors that relays in event motion compensation to detect events triggered by moving obstacles, and uses a potential field approach to execute the evasive maneuver. These methods were designed for quadrotors and are not suitable for ornithopters due to the strong differences between both types of platforms.

Table I compares our approach and the methods in [8]–[10]. First, none of these methods is designed for platforms that move at medium-high velocities. They were validated in quadrotors hovering (or moving up to 1.5 m/s in the case of [10]), in which the majority of the triggered events are caused by the obstacle's motion –simplifying obstacle detection. Our method has been designed for flapping-wing robots which require a minimum flight speed of 3 m/s to flight [1] and suffer from vibrations and angular and linear velocities [4], which trigger additional events caused by the static background, requiring specific moving obstacle detection methods. Moreover, methods [8]–[10] require significantly high onboard weight and computational resources, which could not be mounted on a large-scale ornithopter. Methods [9] and [10] use a powerful embedded computer, a flight board, and an autopilot. Besides, the three methods use two event cameras, which also increases their computational requirements. Although [10] includes a solution with a monocular camera, it uses an additional board to run vision-based state estimation. Conversely, our method uses one only event camera and runs in a single lightweight onboard computer with low computational capacity to satisfy the ornithopter payload restrictions while providing a fast response (control loop closing at 250 Hz) to deal with the high flight velocities. Finally, [9] and [10] segment moving objects by processing *event images* resulting from accumulating the incoming events. Hence, on these methods event processing starts after the events have been accumulated, resulting in delays between event generation and processing, even in cases in which *event image* processing times are lower than the *event image* accumulation times, such as in [10]. Conversely, our method adopts *event-by-event* processing exploiting the asynchronous nature of event generation and enabling shorter

	Flight speeds	**Proc. Weight**	**# of cam**	**C/GPU cores**	**Proc.**	**Accum.**	**Proc. Time**
[8]	∼0 m/s	∼49 g	2	4/0	*e-by-e*	Async	–.
[9]	∼0 m/s	∼150 g	2	6/1	*e-images*	30 ms	12 ms/*e-images*
[10]	[0,1.5] m/s	∼150 g	1-2	6/1	*e-images*	10 ms	3.56 ms/*e-images*
Ours	[2,4] m/s	28.5 g	1	6/0	*e-by-e*	Async.	4 ms/package

TABLE I: Comparison of our approach with other dynamic obstacle avoidance methods –for quadrotors. *e-by-e* regards to *event-by-event*, while *e-images*, to *event images*.

obstacle detection times, which is interesting in our case due to the medium-high ornithopter flight velocities.

III. General Description

Sense-and-avoid of agile aerial robots such as ornithopters requires low latency perception for fast obstacle detection. Event cameras provide visual information with μs resolution triggered by changes of illumination. These sensors have a high dynamic range ($\sim$120 dB) and are robust to illumination conditions and to motion blur, typically desired features in aerial robotics [22]. Previous works have explored the advantages of event cameras in ornithopters [23] [24] and agile quadrotors [20]. Event cameras are compact, have moderate weight, and report low-energy consumption.

The development and integration of *sense-and-avoid* systems for ornithopters entail additional requirements to those considered in other UAVs such as multirotors. First, ornithopters have strict payload capacity and space restrictions which limit the installation of powerful processing hardware, mechanical stabilizers (e.g., gimbals), and sensors. Thus, onboard hardware is carefully selected to satisfy payload and weight balance restrictions and enable real-time onboard processing. Moreover, flapping-wing robots present complex kinematics and dynamics. They are non-holonomic robots using few actuators to control their position and orientation. They generate lift and thrust by flapping their wings, also producing forward, backward, or lateral movements. These aspects set additional requirements for obstacle avoidance including low latency obstacle detection and evasion strategies that consider the platform kinematics and dynamics.

The general diagram of the proposed *sense-and-avoid* scheme is shown in Fig. 2. The *Dynamic Obstacle Motion Estimation* method, see Section IV-A, detects dynamic obstacles and estimates their motion in the image plane. It uses the spatio-temporal information of events triggered by dynamic objects moving with a different velocity than the background. Although many motion-based segmentation methods using traditional framed cameras have been proposed [25], the use of event-based vision has interesting advantages in our problem. First, event-based processing provides natural robustness against motion blur and changes in lighting conditions. Further, motion segmentation using framed cameras process full frames, while event-based methods only analyse asynchronous events enabling faster processing, 250 Hz in the experiments shown in Section V.

The *Collision Risk Evaluation Strategy* module evaluates the risk of collision considering the robot geometry, see Section IV-B. If a collision risk situation is detected an avoidance maneuver is performed by the ornithopter. The optical flow of detected obstacles is used to reactively command the ornithopter to avoid collisions. The adopted *Tail Control* method meets both robustness and simplicity requirements. It actuates on the vertical and horizontal tail deflections to change the ornithopter flight direction. The value of the control commands is adjusted using a simplification of the robot model to consider the dynamic constraints enclosed in the evasion maneuver. The event processing method

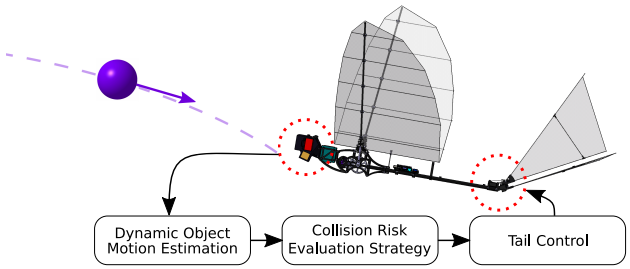

Fig. 2: General diagram of the proposed event-based *sense-and-avoid* scheme for flapping-wing flight.

leverages ASAP [26], which adapts event packaging such that events are processed as soon as possible while avoiding computational overflows, and ensures control closing at 250 Hz with onboard resource-constrained hardware.

IV. Methods

A. Dynamic Object Motion Estimation

The proposed method performs *event-by-event* processing to exploit the asynchronous nature of event cameras. Each event is defined by the tuple $e = (\mathbf{x}, ts, p)$, where $\mathbf{x}$ represents the pixel coordinates (u, v), ts is the timestamp of the event and p is the polarity either positive or negative. The block diagram of the proposed method is shown in Fig. 3. The *Time Filter* module detects events belonging to moving objects using as reference the timestamp of the current and previous events. The event-based *Corner Detector* module finds relevant features from the events triggered by dynamic objects. The *Optical Flow* module determines the direction of motion of the corners belonging to moving objects. Optical flow is computed only from corner events to reduce the computational cost. Finally, *Clustering* gathers optical flow measurements to compute the average flow of the detected dynamic objects. Algorithm 1 describes the proposed dynamic obstacle motion estimation method.

Previous event-based methods for dynamic obstacle detection rely either on optimization methods [27] or motion compensation techniques [10] to distinguish events belonging to moving objects. Our approach focuses on detecting events from dynamic objects performing low computational processing and low latency response suitable for *sense-and-avoid* onboard ornithopters. The *Time Filter* module detects events generated from moving objects using as reference the timestamp difference of the events triggered at the same pixel location. The surface of active events (SAE) $S \in \mathbb{R}^2$ maps the event coordinates $\mathbf{x}$ with the timestamp ts of the last occurring event at $S(\mathbf{x})$. Thus, S describes a 2D representation of the timestamp evolution of triggered events. Under the assumption that objects move with a high relative velocity w.r.t. the robot, the events triggered by dynamic objects are detected using a time threshold τ. If the time difference Δts between an incoming event e_k and $S(\mathbf{x}_k)$ is lower than τ the event is considered to belong to a moving object. Next, the timestamp of e_k updates S by $S(\mathbf{x}_k) = ts_k$ for the future evaluations. The value of τ depends on the velocity of the robot, defined by $\tau = \alpha\tau_{v_z} + (1-\alpha)\tau_{\omega_x}$, where τ_{v_z} and τ_{ω_x} are the contributions due to the current

linear forward and pitch angular velocities of the robot (v_z and ω_x, respectively), and $\alpha \in [0,1]$ sets the contribution of each velocity component. τ_{v_z} and τ_{ω_x} are defined as follows:

$$\begin{bmatrix}\tau_{v_z}\\ \tau_{\omega_x}\end{bmatrix} = \begin{bmatrix}\tau_H\\ \tau_H\end{bmatrix} - (\tau_H - \tau_L)\begin{bmatrix}(v_z - v_L)(v_H - v_L)^{-1}\\ (\omega_x - \omega_L)(\omega_H - \omega_L)^{-1}\end{bmatrix}, \quad (1)$$

where $[v_L, v_H]$ and $[\omega_L, \omega_H]$ are the typical velocity ranges of the robot flight. The operation range $[\tau_L, \tau_H]$ is empirically selected, see Section V.

Algorithm 1: Event-based moving obstacle detection

Input: $\mathbf{e}_k(\mathbf{x}_k, ts_k, p)$
Output: $(\overline{v_x}, \overline{v_y})$
if $ts_k - S(\mathbf{x}_k) < \tau$ **then** ▷ *Time Filter* evaluation
  updateCurrentSlice($\mathbf{e}_k$)
  if *isCorner*($\boldsymbol{e}_k$) **then** ▷ *Corner Detector*
    $(v_x, v_y) \leftarrow$ computeOpticalFlow()
    $id \leftarrow$ updateClusters($\mathbf{e}_k, v_x, v_y$)
    $(\overline{v_x}, \overline{v_y}) \leftarrow$ getClusterFlow(id)
  end
  updateClusters(ts_k) ▷ Retrieve obstacle flow
end
$S(\mathbf{x}_k) = ts_k$ ▷ Update time reference at $\mathbf{x}_k$
$d, \tau_c \leftarrow$ sliceRotation($\mathbf{e}_k$)

Next, the events belonging to dynamic obstacles are processed to estimate their direction of motion, module *Optical Flow* in Fig. 3. That optical flow provides the relative motion estimation between the camera and the objects. Our approach uses the event-based optical flow method ABMOF [28]. It is based on block matching operations between event slices. Slices are 2D histograms of events collected during the accumulation time d. ABMOF includes two different control strategies to vary d depending on the event generation through time. Despite using slices of accumulated events to compute optical flow, it performs *event-by-event* processing by computing the flow of each incoming event. Our method integrates an adapted version of ABMOF to reduce the onboard processing. First, event slices are fed only with events triggered from moving objects. This reduces the computational load by processing only $<32\%$ of the event stream in the experiments of Section V. Second, the optical flow is computed only from events considered as corners. The *eFast event *Corner Detector* in [29] is selected for this task given its fast response and low False Positive Rate. Computing optical flow only from corners reduces the computational cost in $>25\%$ while providing a stable optical flow estimation as described in Section V.

Finally, the resulting event optical flow estimations are clustered to obtain an approximation of the object's optical flow. For this task, we used an adapted version of the *event-by-event* clustering algorithm described in [30]. This algorithm clusters events with spatio-temporal continuity within an adaptive time window. The algorithm was modified to cluster optical flow from events. It receives as input the tuple $f = (\mathbf{x}, ts, \mathbf{v})$, where $\mathbf{v} = (v_x, v_y)$ represents the optical flow estimation of the event with pixel coordinates $\mathbf{x}$ and timestamp ts. Clustering is performed by evaluating the proximity of each new event to a randomly selected event from each of the previous clusters. Each cluster is defined by its centroid $\overline{\mathbf{x}}$, which represents the average location of cluster events, the average optical flow $\overline{\mathbf{v}} = (\overline{v_x}, \overline{v_y})$, and the list Φ of previous tuples f assigned to the cluster. Each new valid optical flow estimation updates $\overline{\mathbf{v}}$ as in Eq. 2.

$$\overline{\mathbf{v}} := \frac{\eta}{\eta+1}\overline{\mathbf{v}} + \frac{1}{\eta+1}\mathbf{v}, \quad (2) \qquad \overline{\mathbf{v}} := \frac{\eta}{\eta-1}\overline{\mathbf{v}} - \frac{1}{\eta-1}\mathbf{v}_\dagger, \quad (3)$$

where η is the number of assigned tuples to the cluster (i.e., the length of Φ). Similarly, the influence of old samples is removed from $\overline{\mathbf{v}}$, see Eq. 3, where $\mathbf{v}_\dagger$ corresponds to flow samples with timestamps lower than τ_c. The parameter τ_c defines the maximum lifetime of flow samples in the cluster and it is dynamically adjusted using the reference feedback from [28]. Finally, ASAP is used to prevent processing overflows by dynamically adapting event packaging to keep the responsiveness of the method.

B. Collision Risk Evaluation Strategy

A reactive evasive maneuver strategy is used to prevent collision risk situations with incoming obstacles. The possible collisions with detected obstacles are determined by considering the geometry of the robot. The ornithopter is approximated by a $2W \times 2H$ safety volume and the obstacle is approximated by a sphere of radius R' (see Fig. 5). R' encloses the obstacle volume and a *Safety Distance* b considering small sensing uncertainties, i.e., R'=R+b. Thus, the minimum angles ψ and θ to avoid a possible collision risk without deviating the flight trajectory are:

$$\psi^*(t) = \arctan\frac{W + 2R'}{z(t) - 2R'}, \quad \theta^*(t) = \arctan\frac{H + 2R'}{z(t) - 2R'}, \quad (4)$$

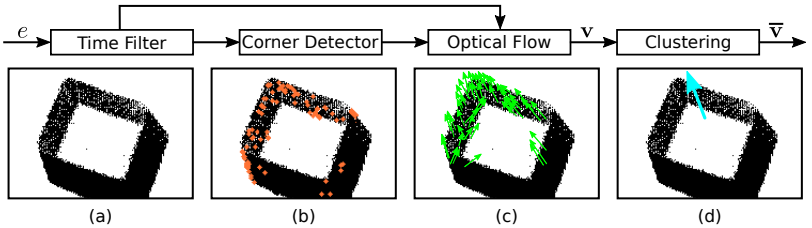

Fig. 3: Block diagram of the *Dynamic Object Motion Estimation* method: a) *Time Filter* detects events triggered by moving objects; b) *Corner Detector* finds relevant features of the object; c) *Optical Flow* estimates the motion of events in the image plane; and d) *Clustering* estimates the object flow using the flow of events belonging to the dynamic object. For clearer visualization the results are shown in *event images* accumulating events every 10 ms.

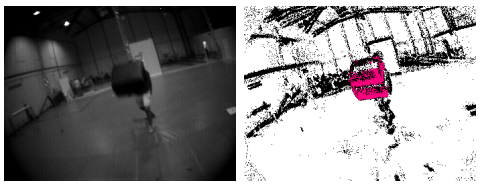

Fig. 4: Event-based dynamic object detection: right) frame from the APS sensor of the DAVIS 346; left) *event image* of events accumulated during 10 ms (in black) including events detected to belong to the moving object (magenta).

where $z(t)$ is the obstacle depth w.r.t. the camera. For any pair of angles (ψ, θ) such that $|\psi(t_1)| \leq |\psi^*(t_1)|$ and $|\theta(t_1)| \leq |\theta^*(t_1)|$ a possible collision between the robot and the obstacle might occur at $t \geq t_1$ if the robot trajectory is not modified. In these cases, an evasive maneuver is activated by guiding the ornithopter away from the obstacle trajectory. The collision evaluation of Eq. 4 depends of the robot-obstacle depth $z(t)$ and its size. Three evaluation cases are considered based on the obstacle available information:

- Assuming $z(t)$ is directly measurable (e.g., using a time-of-flight sensor) and R is known, the collision risk evaluation is directly implemented using Eq. 4.
- Assuming only R is known, $z(t)$ can be estimated by $z(t) = \lambda R / L(t)$, where λ is the camera focal length, and $L(t)$ is the largest side of a rectangle enclosing the clustered events. The value of $L(t)$ is computed by analysing the spatial distribution of events in the cluster.
- If neither $z(t)$ nor R are known, the collision risk cannot be predicted. Thus, any detected obstacle triggers an evasive maneuver.

The second case, having prior information of the object geometry to perform collision risk evaluation, is experimentally validated in Section V. We adopt a conservative strategy that detects collision risk situations using geometrical considerations and exerts reactive evasion maneuvers using the information of the obstacle motion. This reduces processing requirements, enabling low-latency execution.

C. Tail Control

Flapping-wing robots present more complex dynamics than multirotors and fixed-wing robots. Their non-holonomic underactuated nature requires considering the robot dynamic restrictions to evaluate the effect of the actuation commands in future spatial configurations. Conversely, reactive obstacle avoidance requires a fast and robust response to perform aggressive movements. Evasion maneuvers must be performed as fast as possible, thus the control method must be computationally simple. Additionally, since the robot is solely controlled using onboard perception –which implies high levels of uncertainty, obstacle avoidance control must be robust to minimize their effect.

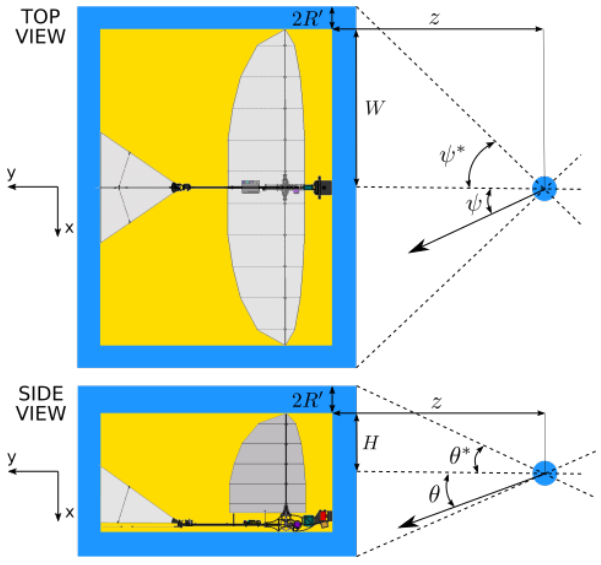

Fig. 5: Collision risk evaluation based on geometry constraints. ψ^* and θ^* are used to determinate collision risk.

One feasible control strategy is to perform an avoidance maneuver with an opposite direction velocity vector w.r.t. the incoming obstacle velocity. From the obstacle mean optical flow $\overline{\mathbf{v}}$ estimated in Section IV-A, the controller computes the longitudinal, δ_e, and lateral, δ_r, tail deflections to perform the evasive maneuver. To accomplish both robust and computationally efficient requirements, the following control law is implemented:

$$\mathbf{u_{react}} = -(\boldsymbol{\kappa_0} + \boldsymbol{\kappa_1}\|\overline{\mathbf{v}}\|) \circ \overline{\mathbf{v}}, \tag{5}$$

where $\circ$ denotes the Hadamard product, $\mathbf{u_{react}} = [\delta_e^{react}, \delta_r^{react}]^T$ is the control action, and $\boldsymbol{\kappa}_0$ and $\boldsymbol{\kappa}_1$ are controller gains. The values of $\boldsymbol{\kappa}_0$ and $\boldsymbol{\kappa}_1$ were experimentally tuned. The followed criterion was to achieve fast control response when an obstacle is detected.

The previous solution assumes that the best maneuver to avoid an obstacle is to fly in the opposite direction to its velocity. However, this strategy is agnostic to the robot dynamics. To cope with this limitation, our method also considers a flapping-wing linearized model adapted from [31] to compute the tail deflections. The model parameters were initially approximated empirically and then fitted by performing different tests inside a motion capture system. The ornithopter linear and angular velocities and its attitude are estimated using an onboard inertial navigation system.

From the current robot configuration, and given $\mathbf{u_{react}}$ from Eq. 5, the adapted model is used to compute the robot acceleration for each tail angle in a discretized range between $\delta_e^{react} \pm 10°$ for longitudinal deflection and between $\delta_r^{react} \pm 10°$ for lateral deflection. The selected tail angle increments, $\mathbf{\Delta u_{model}} = [\Delta\delta_e^{model}, \Delta\delta_r^{model}]^T$, are those that give the robot the greatest acceleration –i.e., those that increase the speed of the evasion maneuver the most in the shortest time. Hence, the tail deflections to command are composed as $\mathbf{u} = \mathbf{u_{react}} + \mathbf{\Delta u_{model}}$. Additionally, our method constrains the deflection values considering the restrictions presented in [32] to avoid stall.

V. Experiments

The experimental platform is the *E-Flap* robot, a customized ornithopter developed at the GRVC Robotics laboratory. The robot has a total length of 95 cm, a wingspan of 1.5 m, a weight of 510 g, and a maximum payload of 520 g at the expense of limited maneuverability and reduced flight time. The sensors and hardware are placed along the robot body to improve the platform maneuverability. A low-cost Khadas VIM3 handles onboard online perception and control. It equips a VectorNav VN-200 inertial navigation system that provides measurements of the ornithopter's body frame velocity. A DAVIS346 event camera provides low latency polarized events from its integrated dynamic vision

Fig. 6: Different size objects considered for the experiments: left) Small Box; center) Stuffed Toy; and right) FitBall.

sensor DVS. The camera weight was reduced to a third of its original value to satisfy the robot weight restrictions. The camera mounts a lens with a total weight of 5 g, a Field of View of 68° horizontal and 53.5° vertical, and an IR-Cut filter to cope with the IR emissions from the motion capture system. The event-based obstacle detector method, the evasive maneuver strategy and the flapping-wing controller avoidance were implemented in C++ using *ROS*. In all the experiments, ASAP was configured to provide event packages at 250 Hz to cope with the low computational restrictions. Thus, the mean optical flow was updated each 4 ms. It is worth mentioning that the *event-by-event* property of our algorithm allows configuring the optical flow update by changing the rate of the packages provided by ASAP.

The proposed dynamic obstacle *sense-and-avoid* system was experimentally validated in both indoor and outdoor scenarios. The GRVC Testbed is a closed area of $15 \times 21 \times 8$ m with 24 *OptiTrack Prime*^{x} 13 cameras providing millimeter accuracy pose estimations. The outdoor scenario is a Soccer Field of 48×54 m with surrounding obstacles suitable to perform short flight experiments with the ornithopter. For all the experiments, the parameters in Eq.1 were set to v_L=3 m/s, v_H=6 m/s, ω_L=1.3 rad/s, and ω_H=3 rad/s, given by the typical flight kinematics conditions of the robot. Further, α was set to 0.8, as the robot mainly performs forward motions. Parameter τ sets the threshold to distinguish the events triggered by dynamic objects. A large value entails detections at longer distances while permitting events triggered by static objects. Besides, a small value of τ performs a quite selective filtering at the expense of reducing the distance at which the obstacles are detected. Hence, τ defines a trade-off between filtering events triggered by static objects and the maximum distance to detect an obstacle. From our experiments, τ_L=15 ms and τ_H=25 ms were empirically selected to provide a trade-off between having detection distances of 6 m while filtering 82% of events triggered by the scene background.

The experimental validation was divided into two parts. First, the dynamic obstacle detection and motion estimation were evaluated. Second, the evaluation of the full obstacle avoidance system was performed in experiments in indoor and outdoor scenarios with different illumination conditions.

A. Obstacle detection and motion estimation evaluation

In these experiments, obstacles were thrown into the Field of View (FoV) of the camera while the ornithopter performed forward flight. The experiments were performed in the Testbed scenario to retrieve the pose information from the obstacle and the robot. The goal was to evaluate the detection and motion estimations performance of the method described in Section IV-A. Three objects with different sizes were considered (Fig. 6): a Small Box of size 220 × 200 × 150 mm, a Stuffed Toy of size 400 × 450 × 400 mm, and a Fitball with a diameter of 750 mm. Obstacle detection was evaluated in distances ranging from 0.5 m to 6 m. At distances closer than 0.5 m the obstacle body filled a large zone of the image leading to invalid detections. At large distances obstacles were represented by few pixel events due to the camera resolution, which hindered their 2D representation for obstacle detection. A total of 45 experiments were performed with each obstacle.

The detection performance was evaluated by comparing the outcome of our algorithm with the frames provided by the APS sensor. The centroid of the detected moving object was rendered into *event images* obtained each 25 ms. The distance between the object's centroid in both frames was used as evaluation criteria. Each comparison led to a possible result: False Positive (FP), False Negative (FN), True Positive (TP), and True Negative (TN). A True Positive occurred when the Manhattan distance between the centroid of both objects was $\leq \eta$ pixels. Fig. 7 shows a histogram of the distance between the object and the ground of truth in the performed experiments. The majority of samples at distances >15 px correspond to False Positives. Choosing η to 10 was a suitable trade-off between validating the majority valid samples as True Positives while rejecting False Positive samples. Next, the overall Accuracy, Precision, True Positive Rate (TPR), and False Positive Rate (FPR) were computed. Table II summarizes the detection results of each obstacle at different distances. The method reported an overall accuracy of 91.7%. The distance directly affected the detection performance specially with small objects at distances >4 m. At such distances, small objects triggered few events hampering object detection, and the lack of triggered events affected the obstacle detection as many events in $S(\mathbf{x})$ did not satisfy the τ condition. In general, at distances between 2 to 4 m the method reported an accuracy above 94.7%. In this range, the size of the different objects in the image was enough to perform successful dynamic object detection in the majority of the experiments. Additionally, the method provided a low number of False Positives as reported by the average FPR of 4.2%, and an average Precision of 95.1%. Finally, a few detections were missed during the experiments as evidenced by the average TPR of 88.5%.

The obstacle detection method was evaluated in multi-obstacle experiments in which three objects were thrown from different directions in the camera FoV at similar times. It reported an accuracy of 74.2% among all the experiments. The performance reduction was caused by the generation of False Positive samples when the obstacles overlapped in the image plane, which tended to merge clusters hampering the individual detection of objects. Despite this degradation, the method results were satisfactory taking into account that the detection was performed with a single event camera.

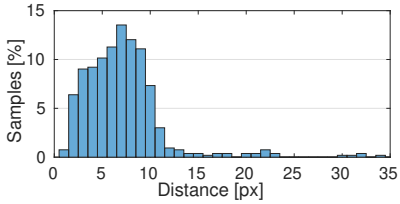

Fig. 7: Histogram of the distance between the detected object and the Ground Truth in the overall tests.

	1.5 – 2.0 m				**2.0 – 4.0 m**				**4.0 – 6.0 m**			
	Acc	*Pre*	*TPR*	*FPR*	*Acc*	*Pre*	*TPR*	*FPR*	*Acc*	*Pre*	*TPR*	*FPR*
Small Box	0.91	0.95	0.86	0.04	0.97	0.98	0.97	0.03	0.80	0.86	0.75	0.09
Stuffed Toy	0.97	0.95	0.97	0.04	0.93	0.98	0.86	0.02	0.92	0.97	0.85	0.02
Fitball	0.91	0.98	0.82	0.01	0.94	0.94	0.93	0.05	0.92	0.94	0.91	0.08

TABLE II: Accuracy, Precision, True Positive Rate (TPR), and False Positive Rate (FPR) results of dynamic obstacle detection using the different obstacles while varying the launching distance.

The motion estimation evaluation consisted of comparing the mean optical flow direction of the dynamic obstacle on the image plane with the ground truth direction. The ground truth was obtained from the motion capture system by projecting the position of the obstacle in the image and estimating its motion direction from previous and future samples. For better validation, the obstacles were launched from a distance of 8 m to describe larger trajectories. A total of 30 experiments were performed. The error was defined as the instantaneous angle difference between the estimated direction of the obstacle movement and the ground truth direction. Fig. 8 shows the quartiles, mean, and standard deviation of the error along each experiment. The resulting Root Mean Square Error was $11.2°$, which is a reasonable error to guide the robot in a collision-free direction. *Safety Distance* b described in Section IV-B was used to consider this error by enlarging the obstacle geometry to add extra safety to the evasion maneuver.

B. Sense-and-avoid evaluation

Next, the full proposed *sense-and-avoid* system was evaluated. Two cases can be distinguished in case the obstacle detection method fails. A False Positive obstacle detection triggers an unnecessary evasion maneuver. A False Negative obstacle detection neglects the evaluation of a collision risk situation, potentially causing an impact between the robot and the obstacle. Different sets of experiments were analysed to evaluate its performance. The ornithopter was launched towards a goal zone by an operator and performed forward flight using the controller described in [5]. One of the obstacles in Fig. 6 was launched to intercept the ornithopter. The obstacles were launched in various directions to evaluate different evasive maneuvers. To enhance repeatability the lightweight obstacles were launched using a motorized launcher which set their initial speed to a specific value. Obstacles were thrown from a distance of 10 m and moved with an average speed in the range of $[5, 8]$ m/s. Our system checked for collision risk situations and activated an avoidance maneuver if a collision risk situation was detected. Three types of analyses were performed.

First, we performed experiments to analyse the performance of the system in case an *Intersection of the Safety Volumes (ISV)* occurred. An *ISV* exists when the robot safety volume and the sphere of radius R' enclosing the object intersect along the robot trajectory in case it performs no evasion maneuver. These experiments were performed indoors: the ground truth robot and obstacle positions were recorded using a motion caption system to check if an *ISV* occurred. A total of 25 experiments were performed with each object using regular (760 lx) and dark (<15 lx) illumination conditions. The average avoidance success rate with the three objects, see Table III, was 90.7%. The best result was obtained using the FitBall as obstacle, which larger size allowed an earlier detection of the collision risk situation. The experiments with dark illumination conditions also reported a remarkable success rate: the slight performance degradation was caused by the additional noisy events produced by the poor lighting.

Second, we performed experiments to analyze the system performance in case *ISV* situations did not occur. 20 experiments were performed using the Small Box obstacle. In 85% of the experiments, the system did not detect collision risk situations. Only in 15% of the cases, it detected collision risk situations –activating an unnecessary evasive maneuver– in flights with no *ISV* situations. This result was mainly caused by the conservative selection of the radius R' which enlarged the obstacle size to reduce impacts with the robot body. False

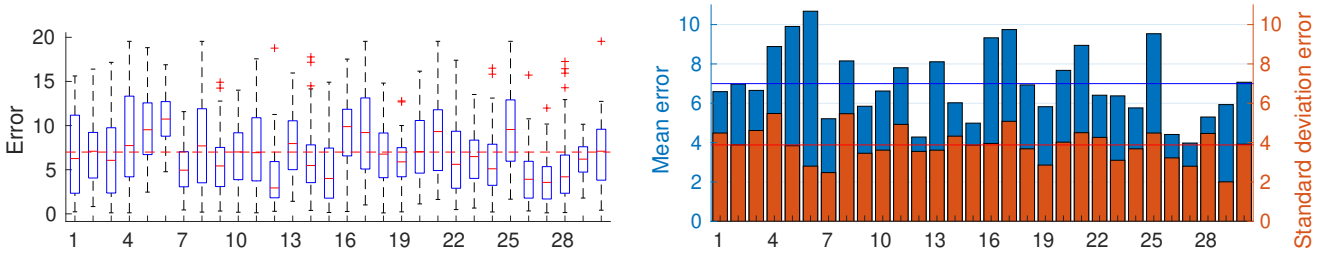

Fig. 8: Box plot (left) and mean and standard deviation (right) of the motion direction error along each of 30 experiments. The direction error is defined as the instantaneous absolute difference between the direction of motion estimated by our approach and the obstacle direction from the ground truth. The mean error considering all the experiments is $6.97°$. The mean standard deviation considering all the experiments is $3.89°$, and maximum instantaneous direction error is $19.54°$.

Positive detections increased in the dark lighting experiments due to the higher level of noisy events.

Finally, the system performance in outdoor experiments was analyzed to evaluate its robustness to different scenarios. In these experiments, the collisions were evaluated visually due to the lack of motion capture information. A total of 25 experiments were performed with each object under light and dark lighting conditions. The results are shown in Table III. The proposed system had a success rate of 92.0% with regular illumination conditions, and reported acceptable results with dark lighting conditions 85.3%.

	Validation	Small Box	Stuffed Toy	FitBall
Indoors	Motion Capture Sys.	92%	92%	96%
Indoors Dark		84%	88%	92%
Outdoors	Visual	92%	92%	92%
Outdoors Dark		84%	84%	88%

TABLE III: Success rate of the proposed dynamic obstacle avoidance method in different scenarios.

VI. Conclusions and Future Work

This paper presents the first event-based obstacle avoidance system for large-scale flapping-wing robots. The proposed approach exploits the advantages of event-based vision to detect dynamic obstacles and perform evasion maneuvers. Our scheme has been validated in several indoor and outdoor scenarios with different illumination conditions. It reports an average avoidance success rate of 89.7% evading dynamic obstacles of different sizes and shapes. Further, its *event-by-event* processing nature and efficient implementation allow fast onboard computation even in low processing capacity hardware, providing high rate estimations (250 Hz in our experiments). Future work includes the validation of our method in other agile robots. Further, the development of a map-based method for the avoidance of static obstacles and its integration in a complete obstacle avoidance system for static and dynamic objects are object of future research.

References

[1] R. Zufferey, J. Tormo, M. Guzman, F. Maldonado, E. Sanchez-Laulhe, P. Grau, M. Perez, J. Á. Acosta, and A. Ollero, "Design of the high-payload flapping wing robot e-flap," *IEEE Robot. Auto. Lett.*, 2021.

[2] G. de Croon, "Flapping wing drones show off their skills," *Sci. Rob.*, vol. 5, 2020.

[3] N. Haider, A. Shahzad, M. N. Mumtaz Qadri, and S. I. Ali Shah, "Recent progress in flapping wings for micro aerial vehicle applications," *Proc. Inst. Mech. Eng., Part C: J. of Mech. Eng. Sci.*, 2020.

[4] A. Gómez Eguíluz, J. P. Rodríguez-Gómez, J. Paneque, P. Grau, J. R. Martinez De-Dios, and A. Ollero, "Towards flapping wing robot visual perception: Opportunities and challenges," in *IEEE RED-UAS*, 2019.

[5] F. Maldonado, J. Acosta, J. Tormo-Barbero, P. Grau, M. Guzmán, and A. Ollero, "Adaptive nonlinear control for perching of a bioinspired ornithopter," in *IEEE/RSJ IROS*, 2020, pp. 1385–1390.

[6] E. Farrell Helbling and R. J. Wood, "A review of propulsion, power, and control architectures for insect-scale flapping-wing vehicles," *App. Mech. Review.*, vol. 70, no. 1, pp. 1–9, 2018.

[7] G. Gallego, T. Delbruck, G. M. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. Davison, J. Conradt, K. Daniilidis, and D. Scaramuzza, "Event-based vision: A survey," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2020.

[8] E. Mueggler, N. Baumli, F. Fontana, and D. Scaramuzza, "Towards evasive maneuvers with quadrotors using dynamic vision sensors," in *European Conf. on Mobile Robots (ECMR)*, 2015, pp. 1–8.

[9] N. J. Sanket, C. M. Parameshwara, C. D. Singh, A. V. Kuruttukulam, C. Fermuller, D. Scaramuzza, and Y. Aloimonos, "Evdodgenet: Deep dynamic obstacle dodging with event cameras," in *ICRA*, 2020, pp. 10 651–10 657.

[10] D. Falanga, K. Kleber, and D. Scaramuzza, "Dynamic obstacle avoidance for quadrotors with event cameras," *Sci. Rob.*, vol. 5, no. 40, 2020.

[11] S. Hrabar, "Reactive obstacle avoidance for rotorcraft uavs," in *IEEE/RSJ IROS*, 2011, pp. 4967–4974.

[12] H. Oleynikova, D. Honegger, and M. Pollefeys, "Reactive avoidance using embedded stereo vision for mav flight," in *ICRA*, 2015.

[13] D. Falanga, S. Kim, and D. Scaramuzza, "How fast is too fast? the role of perception latency in high-speed sense and avoid," *IEEE Robot. Autom. Lett.*, vol. 4, no. 2, pp. 1884–1891, 2019.

[14] Z. Ma, C. Wang, Y. Niu, X. Wang, and L. Shen, "A saliency-based reinforcement learning approach for a uav to avoid flying obstacles," *Robot. Auto. Syst.*, vol. 100, pp. 108–118, 2018.

[15] F. Garcia Bermudez and R. Fearing, "Optical flow on a flapping wing robot," in *IEEE/RSJ IROS*, 2009, pp. 5027–5032.

[16] G. de Croon, E. de Weerdt, C. de Wagter, and B. Remes, "The appearance variation cue for obstacle avoidance," in *IEEE ICRB*, 2010, pp. 1606–1611.

[17] S. Tijmons, G. C. H. E. de Croon, B. D. W. Remes, C. De Wagter, and M. Mulder, "Obstacle avoidance strategy using onboard stereo vision on a flapping wing mav," *IEEE Trans. Robot.*, vol. 33, no. 4, pp. 858–874, 2017.

[18] B. J. Pijnacker Hordijk, K. Y. Scheper, and G. C. De Croon, "Vertical landing for micro air vehicles using event-based optical flow," *J. of Field. Robot.*, vol. 35, no. 1, pp. 69–90, 2018.

[19] A. Gómez Eguíluz, J. P. Rodríguez-Gómez, J. R. Martínez-de Dios, and A. Ollero, "Asynchronous event-based line tracking for time-to-contact manuevers in UAS," in *IEEE/RSJ IROS*, 2020, pp. 5978–5985.

[20] J. Delmerico, T. Cieslewski, H. Rebecq, M. Faessler, and D. Scaramuzza, "Are we ready for autonomous drone racing? the UZH-FPV drone racing dataset," in *IEEE ICRA*, 2019, pp. 6713–6719.

[21] S. Sun, G. Cioffi, C. de Visser, and D. Scaramuzza, "Autonomous quadrotor flight despite rotor failure with onboard vision sensors: Frames vs. events," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 580–587, 2021.

[22] J. P. Rodríguez-Gómez, A. G. Eguíluz, J. R. Martínez-De Dios, and A. Ollero, "Auto-tuned event-based perception scheme for intrusion monitoring with uas," *IEEE Access*, vol. 9, pp. 44 840–44 854, 2021.

[23] J. Rodríguez Gómez, R. Tapia, J. Paneque, A. Gómez Eguíluz, P. Grau, J. Martínez de Dios, and A. Ollero, "The GRIFFIN perception dataset: Bridging the gap between flapping-wing flight and robotic perception," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 1066–1073, 2021.

[24] A. G. Eguiluz, J. Rodriguez-Gomez, R. . Tapia, J. Maldonado, J. Acosta, J. M. de Dios, and A. Ollero, "Why fly blind? event-based visual guidance for ornithopter robot flight," in *IEEE/RSJ IROS*, 2021, pp. 1935–1942.

[25] Z. Dengsheng and L. Guojun, "Segmentation of moving objects in image sequence: A review," *Circt., Syst. Sign. Proc.*, vol. 20, no. 2, pp. 143–183, 2001.

[26] R. Tapia, A. Gómez Eguíluz, J. Martınez-de Dios, and A. Ollero, "ASAP: Adaptive scheme for asynchronous processing of event-based vision algorithms," in *IEEE ICRA Work. Unconv. Sens. Robot.*, 2020.

[27] A. Mitrokhin, C. Fermüller, C. Parameshwara, and Y. Aloimonos, "Event-based moving object detection and tracking," in *IEEE/RSJ IROS*, 2018, pp. 6895–6902.

[28] M. Liu and T. Delbruck, "Adaptive time-slice block-matching optical flow algorithm for dynamic vision sensors," in *British Machine Vision Conference (BMVC) 2018*. Proc. of BMVC 2018, September 2018.

[29] E. Mueggler, C. Bartolozzi, and D. Scaramuzza, "Fast event-based corner detection." in *British Machine Vision Conf. (BMVC)*, 2017.

[30] J. P. Rodríguez-Gómez, A. Gómez Eguíluz, J. R. Martínez-de Dios, and A. Ollero, "Asynchronous event-based clustering and tracking for intrusion monitoring in UAS," in *IEEE ICRA*, 2020.

[31] S. Armanini, C. De Visser, G. Croon, and M. Mulder, "Time-varying model identification of flapping-wing vehicle dynamics using flight data," *J. Guidan. Contr. Dynamic.*, vol. 39, 03 2016.

[32] M. Guzmán, C. R. Páez, F. J. Maldonado, R. Zufferey, J. Tormo-Barbero, J. Acosta, and A. Ollero, "Design and comparison of tails for bird-scale flapping-wing robots," in *IEEE/RSJ IROS*, 2021, pp. 6335–6342.