2020 IEEE International Conference on Robotics and Automation (ICRA)
31 May - 31 August, 2020. Paris, France

EVDodgeNet: Deep Dynamic Obstacle Dodging with Event Cameras

Nitin J. Sanket[1], Chethan M. Parameshwara[1], Chahat Deep Singh[1], Ashwin V. Kuruttukulam[1], Cornelia Fermüller[1], Davide Scaramuzza[2], Yiannis Aloimonos[1]

***Abstract*— Dynamic obstacle avoidance on quadrotors requires low latency. A class of sensors that are particularly suitable for such scenarios are event cameras. In this paper, we present a deep learning based solution for dodging multiple dynamic obstacles on a quadrotor with a single event camera and on-board computation. Our approach uses a series of shallow neural networks for estimating both the ego-motion and the motion of independently moving objects. The networks are trained in simulation and directly transfer to the real world without any fine-tuning or retraining. We successfully evaluate and demonstrate the proposed approach in many real-world experiments with obstacles of different shapes and sizes, achieving an overall success rate of 70% including objects of unknown shape and a low light testing scenario. To our knowledge, this is the first deep learning – based solution to the problem of dynamic obstacle avoidance using event cameras on a quadrotor. Finally, we also extend our work to the pursuit task by merely reversing the control policy, proving that our navigation stack can cater to different scenarios.**

Supplementary Material

The accompanying video, supplementary material, code and dataset are available at http://prg.cs.umd.edu/EVDodgeNet

I. Introduction and Philosophy

The never-ending quest to understand and mimic ultra-efficient flying agents like bees, flies, and birds has fueled the human fascination to create autonomous, agile and ultra-efficient small aerial robots. These robots are not only utilitarian but are much safer to operate in static or dynamic environments and around other agents as compared to their larger counterparts. Need for creation of such small aerial robots has given rise to the development of numerous perception algorithms for low latency obstacle avoidance. Here, latency is defined as the time the robot takes to perceive, interpret and generate control commands [1].

Low latency static obstacle avoidance has been studied extensively in the last decade [2]. Recently, however, dynamic obstacle avoidance has gained popularity in the field of robotics due to the exponential growth of event cameras. These are bioinspired vision sensors that output per-pixel temporal intensity differences caused by relative motion with microsecond latency [3].

Event cameras have the potential to become the de-facto standard for visual motion estimation problems due to their inherent advantages of low latency, high temporal resolution, and high dynamic range [4]. These advantages make event cameras tailor made for dynamic obstacle avoidance.

Nitin J. Sanket and Chethan M. Parameshwara contributed equally to this work. (Corresponding author: Nitin J. Sanket.)

[1]Perception and Robotics Group, University of Maryland Institute for Advanced Computer Studies, University of Maryland, College Park.

[2]Robotics and Perception Group, Dep. of Informatics, University of Zurich, and Dep. of Neuroinformatics, University of Zurich and ETH Zurich.

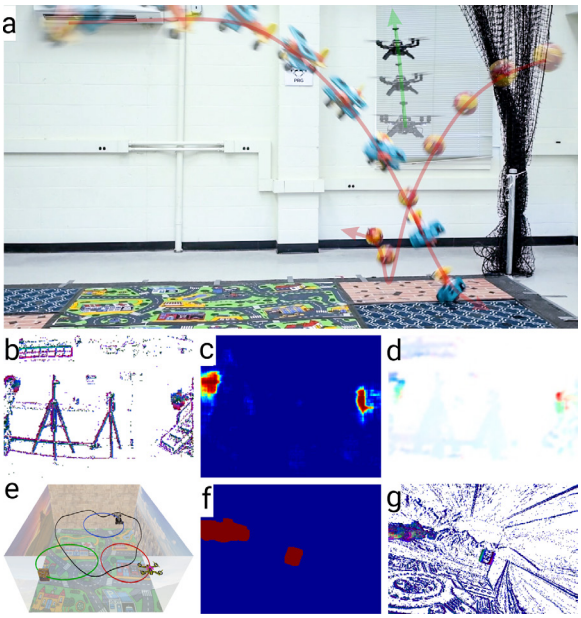

Fig. 1. (a) A real quadrotor running *EVDodgeNet* to dodge two obstacles thrown at it simultaneously. (b) Raw event frame as seen from the front event camera. (c) Segmentation output. (d) Segmentation flow output which includes both segmentation and optical flow. (e) Simulation environment where *EVDodgeNet* was trained. (f) Segmentation ground truth. (g) Simulated front facing event frame. *All the images in this paper are best viewed in color.*

In this paper, we present a framework to dodge multiple unknown dynamic obstacles on a quadrotor with event cameras using deep learning. Although dynamic obstacle detection using traditional cameras and deep learning has been extensively studied in the computer vision community under the umbrellas of object segmentation and detection, they are either of high latency, computationally expensive (not enough to be used on micro/nano-quadrotors) and/or do not generalize to novel objects without retraining or fine-tuning.

Our work is closely related to [1] with the key difference being that our approach uses deep learning and generalizes to unknown real objects after being trained only on simulation.

A. Problem Formulation and Contributions

A quadrotor moves in a static scene with multiple Independently Moving dynamic Objects/obstacles (IMOs). The quadrotor is equipped with a front facing event camera, a downfacing lower resolution event camera coupled with sonar, for altitude measurements, and an IMU.

The problem we address is as follows: *Can we present an AI framework for the task of dodging/evading/avoiding these dynamic obstacles without any prior knowledge, using only on-board sensing and computation?*

We present various flavors of the dodging problem, such as hovering quadrotor dodging unknown obstacles,

978-1-7281-7395-5/20/$31.00 ©2020 IEEE

slow-moving quadrotor dodging unknown shaped obstacles given a bound on size, hovering and slow moving quadrotor dodging known objects (particularly targeted to spherical objects of known radii). We extend our approach by demonstrating pursuit/intercept of a known object using the same deep-learning framework. This showcases that our proposed framework can be used in a general navigation stack on a quadrotor and can be re-purposed for various related tasks. A summary of our contributions are (Fig. 1):

- We propose and implement a network (called EVDeBlurNet) that *deblurs* event frames, such that learning algorithms trained on simulated data can generalize to real scenes without retraining or fine-tuning.
- We design and implement a network (called EVSegFlowNet) that performs both segmentation and optical flow of IMOs to obtain both segmentation and optical flow in a single network.
- We propose a control policy based on estimated motion of multiple IMOs under various scenarios.
- We evaluate and demonstrate the proposed approach on a real quadrotor with onboard perception and computation.

B. Related Work

We subdivide the related work into three parts, i.e., ego-motion estimation, independent motion segmentation, and obstacle avoidance.

1) Independent Motion Detection and Ego-motion Estimation – Two sides of the same coin

Information from complementary sensors, such as standard cameras and Inertial Measurement Units (IMUs), has given rise to the field of Visual Inertial Odometry (VIO) [5], [6]. Low latency VIO algorithms based on event cameras have been presented in [4], [7], which use classical feature tracking inspired methods to estimate ego-motion. Other works, instead, try to add semantic information to enhance the quality of odometry by adding strong priors about the scene [8], [9]. Most works in the literature focus on ego-motion estimation in static scenes which are seldom encountered in the real world. To account for moving objects, these algorithms implement a set of outlier rejection schemes to detect IMOs. We would like to point out that by carefully modelling these "outliers" one can estimate both ego-motion and IMO motion [10].

2) Image stabilization as a key to independent motion segmentation

Keen readers might have contrived that by performing the process of image stabilization IMOs would "stand-out". Indeed, this was the approach most robust algorithms used in the last two decades. A similar concept was adapted in some recent works on event-based cameras for detecting IMOs [11]–[13]. Recently a deep learning based approach was presented for IMO detection using a structure from motion inspired approach [14].

3) Obstacle avoidance on aerial robots

The works presented in the above two subsections have aided the advancement of obstacle avoidance on aerial robots. [15], [16] presented approaches for high speed static obstacle avoidance by estimating depth maps and visual servoing using a monocular imaging camera respectively. [17] provides a detailed collation of the prior work on

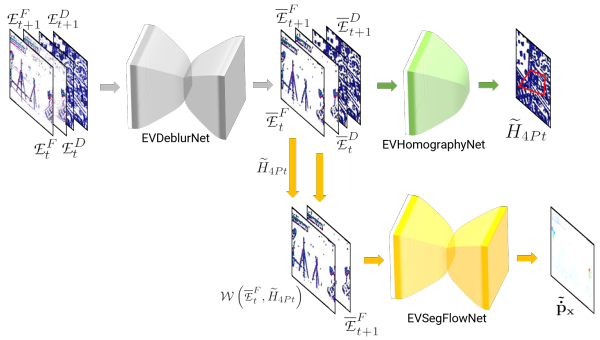

Fig. 2. Overview of the proposed neural network based navigation stack for the purpose of dodging.

static obstacle avoidance using stereo cameras. A hardware and software architecture was proposed in [18], [19] for high speed quadrotor navigation by mapping the cluttered environment using a lidar. Using event cameras for high speed dodging is not new and the first work was presented in [20] where an approach was presented to avoid a dynamic spherical obstacle using stereo event cameras. Very recently, [1] presented a detailed analysis of perception latency for dodging a dynamic obstacle.

C. Organization of the paper

The paper is structured into perception and control modules. The perception module (Refer to Fig. 2) is further divided into three segments.

1. The input to the perception system are event frames (Sec. II-A). Such a projection of event data to generate *event frames* suffers from misalignment [21] unless motion compensation is performed. We call this misalignment or loss of contrast/sharpness as *blur* due to its visual resemblance to classical image motion blur. To perform motion compensation and denoising, we present a neural network called *EVDeBlurNet* in Sec. II-A.
2. Suppl. Sec. S.III. presents how ego-motion is obtained using *EVHomographyNet*.
3. Sec. II-B describes how segmentation and optical flow of IMOs are obtained using the novel *EVSegFlowNet*.

Sec. III presents the control scheme for dodging given the outputs from the perception module. We also bring the generality of our perception stack into limelight in Suppl. Sec. S.IX by adapting our approach to the problem of pursuit. Sec. IV illustrates the experimental setup and provides error analyses of the approaches presented along with detailed ablation studies. We finally conclude the paper in Sec. V with parting thoughts on future work.

II. Deep Learning Based Navigation Stack For Dodging Dynamic Obstacles

Fig. 2 shows an overview of our proposed approach. Refer to Suppl. Sec. S.I for the coordinate frame definitions. Our hardware setup is shown in Fig. 3.

A. EVDeBlurNet

The event frame $\mathcal{E}$ consists of three channels. The first and second channels contain the per-pixel average count of positive and negative events. The third channel contains the per-pixel average time between events (refer to Sec. S.II for a mathematical formulation). Though event representation offers many advantages regarding computational complexity

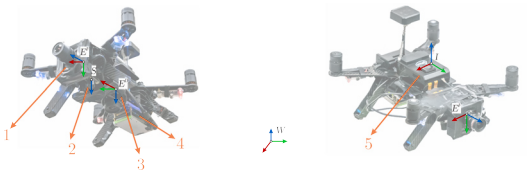

Fig. 3. Representation of coordinate frames on the hardware platform used. (1) Front facing DAVIS 240C, (2) down facing sonar on PX4Flow, (3) down facing DAVIS 240B, (4) NVIDIA TX2 CPU+GPU, (5) Intel® Aero Compute board.

and providing tight time bounds on operation, there is a hitch. Event frames can be "blurry" (projection of misaligned events) based on a combination of the integration time δt (observe in Fig. 6 how sharpness of the image decreases as integration time δt increases), apparent scene movement on the image plane (which depends on the amount of camera movement and depth of the scene) and scene contrast (contrast of the latent image pixels). Here, we define blur on the event frame $\mathcal{E}$ as the misalignment of the events in a small integration time δt.

An event is triggered when the relative contrast (on the latent image I) exceeds a threshold τ and is mathematically modelled as: $\| \log(I) \|_1 \approx \| \langle \nabla_{\mathbf{x}} \log(I), \dot{\mathbf{x}} \Delta t \rangle \|_1 \geq \tau$.

Here, $\langle \cdot, \cdot \rangle$ denotes the inner/dot product between two vectors, $\nabla_{\mathbf{x}}$ is the spatial gradient, $\dot{\mathbf{x}}$ is the motion field on the image and Δt is the time since the previous event at the same location. The above equation elucidates how the latent image contrast, motion and depth are coupled to event frames. Note that, $\dot{\mathbf{x}}$ depends on the 3D camera motion and the scene depth. We refer the reader to [22] for more details.

This "blurriness" of the event frame can adversely affect the performance of algorithms built on them. To alleviate this problem, we need to deblur the event images. This is fairly easy if we directly use the spatio-temporal event cloud and follow the approach described in [21]. Essentially the problem deals with finding point trajectories along the spatio-temporal point cloud to maximize a heuristically chosen contrast function. Mathematically, we want to solve the following problem: $\operatorname{argmax}_{\theta} \mathcal{C}(\mathcal{W}(\mathcal{E}, \theta))$. Here $\mathcal{C}$ is a heuristic contrast function and θ are the parameters of point trajectories in the spatio-temporal point cloud according to which the events are warped and $\mathcal{W}(\mathcal{E}, \theta)$ represents the event image formed by the warped events. In our scenario, we want to model the deblurring problem in 2D, i.e., working on $\mathcal{E}$ directly without the use of a spatio-temporal point cloud so that the problem can be solved efficiently using a 2D Convolutional Neural Network (CNN). Such a deblurring problem using a single image has been studied extensively for traditional cameras for rectifying motion blurred photos. Our modified problem in 2D can be formulated as: $\operatorname{argmax}_{\mathcal{K}} \mathcal{C}(\mathcal{K} \circledast \mathcal{E})$. Here $\mathcal{K}$ is the heterogeneous deblur kernel and $\circledast$ is the convolution operator. However, estimating $\mathcal{K}$ directly is not constrained enough to be learned in an unsupervised manner. Instead, we formulate the deblurring problem inspired by Total Variation (TV) denoising to give us the final optimization problem as follows: $\operatorname{argmax}_{\overline{\mathcal{E}}} \mathcal{C}(\overline{\mathcal{E}}) + \lambda \operatorname{argmin}_{\overline{\mathcal{E}}} \mathcal{D}(\mathcal{E}, \overline{\mathcal{E}})$. Here $\overline{\mathcal{E}}$ represents the deblurred event frame, λ is a regularization penalty and $\mathcal{D}$ represents a distance function to measure similarity between two event frames. Note that directly solving $\operatorname{argmax}_{\overline{\mathcal{E}}} \mathcal{C}(\overline{\mathcal{E}})$ yields trivial solutions of high frequency noise.

To learn the function using a neural network we convert the argmax operator into an argmin operator as follows:

$$\operatorname*{argmin}_{\overline{\mathcal{E}}} \; -\mathcal{C}(\overline{\mathcal{E}}) + \lambda \mathcal{D}(\mathcal{E}, \overline{\mathcal{E}}) \tag{1}$$

Refer to Suppl. Sec. III for a detailed mathematical description of all the loss functions. Intuitively, the higher the value of the contrast, the lower the value of the loss function, but going away too far from the input will penalize the loss function striking a balance between high contrast and similarity to the input image. We call our CNN which generates the deblurred event images *EVDeBlurNet*. It takes as input $\mathcal{E}$ and outputs deblurred $\overline{\mathcal{E}}$. The network architecture is a simple encoder-decoder with four convolutional and four deconvolutional layers with batch normalization (Suppl. Sec. S.V). Another benefit of the encoder decoder's lossy reconstruction is that it removes stray events (which are generally noise) and retains events corresponding to contours, thereby greatly increasing the signal to noise ratio.

Recently, [23] also presented a method for deblurring event frames to improve optical flow estimation via a coupling between predicted optical flow and sharpness in the event frame in the loss function. In contrast, our work presents a problem-independent deblurring network without the supervision from optical flow. We obtain "cheap" odometry using the *EVHomographyNet* as described in Suppl. Sec. S.III which is built upon [24], [25].

B. EVSegFlowNet

The end goal of this work is to detect/segment Independently Moving Objects (IMOs) and to dodge them. One could fragment this problem into two major parts, detecting IMOs, and subsequently estimating their motion to issue a control command to move away from the IMO in a safe manner. Let's start by discussing each fragment. Firstly, we want to segment the object using consecutive event frames $\mathcal{E}_t$ and $\mathcal{E}_{t+1}$. A simple way to accomplish this is by generating simulated data with known segmentation for each frame and then training a CNN to predict the foreground (IMO)/background segmentation. Such a CNN can be trained using a simple cross-entropy loss function: $\operatorname{argmin}_{p_f} -\mathbb{E}(\mathbb{1}_f \log(p_f) + \mathbb{1}_b \log(p_b))$. Here, $\mathbb{1}_f, \mathbb{1}_b$ are the indicator variables denoting if a pixel belongs to foreground or background. They are mutually exclusive, i.e., $\mathbb{1}_f = \neg \mathbb{1}_b$ and p_f, p_b represent the foreground and background predicted probabilities where $p_f + p_b = 1$. Note that each operation in the above equation is performed per pixel, and then an average over all pixels is computed. In the second step we want to estimate the IMO motion. Without any prior knowledge about the IMO it is impossible to estimate the 3D motion of the IMO from a monocular camera (event based or traditional). To make this problem tractable, we assume a prior about the object. More details can be found in Sec. III.

Once we have a prior about the object, we can estimate the 3D IMO motion using optical flow of the pixels corresponding to the IMO on the image plane. A simple way to obtain optical flow is to train a CNN in a supervised manner. However, recent research has shown that these do not generalize well to new scenes/objects [26]. A better way is to use a self-supervised or completely unsupervised loss function: $\operatorname{argmin}_{\dot{\mathbf{x}}} \mathbb{E}(\mathcal{D}(\mathcal{W}(\mathcal{E}_t, \dot{\mathbf{x}}), \mathcal{E}_{t+1}))$. Here $\dot{\mathbf{x}}$ is the estimated optical flow between $\mathcal{E}_t \mapsto \mathcal{E}_{t+1}$ and $\mathcal{W}$ is a differentiable warp function based on optical

flow and bilinear interpolation implemented using an STN. The self-supervised flavor of this algorithm [27] utilizes corresponding image frames instead of event frames for the loss function but the input is still the stack of event frames. One could utilize the two networks we talked about previously and solve the problem of dodging, however, one would need to run two neural networks for this purpose. Furthermore, this method suffers from a major problem: any unsupervised or self-supervised method can estimate rigid optical flow (optical flow corresponding to the background regions $\mathcal{B}$) accurately but the non-rigid optical flow (optical flow corresponding to the foreground regions $\mathcal{F}$) is not very accurate. This is an artifact because of the number of pixels corresponding to the foreground is often far less than that corresponding to the background, i.e., $\overline{\overline{\mathcal{F}}} \ll \overline{\overline{\mathcal{B}}}$. One would have to train for a lot of iterations to obtain accurate optical flow results on these foreground pixels which runs into the risk of overfitting to the dataset. This defeats the promise of self-supervised or unsupervised formulations.

To solve both the problems of complexity and accuracy, we formulate the problem using a semi-supervised approach to learn segmentation and optical flow at the same time, which we call *EVSegFlowNet*. We call the output of the network *segmentation flow* denoted by $\tilde{\dot{\mathbf{p}}}$ which is defined as follows.

$$\tilde{\dot{\mathbf{p}}}_{\mathbf{x}} = \dot{\mathbf{x}}, \text{ if } \mathbb{1}_f(\mathbf{x}) = 1 \text{ and } \tilde{\dot{\mathbf{p}}}_{\mathbf{x}} = \mathbf{0}, \text{ if } \mathbb{1}_b(\mathbf{x}) = 1 \quad (2)$$

One could intuit that we can obtain a noisy segmentation for free by simple thresholding on the magnitude of $\tilde{\dot{\mathbf{p}}}_{\mathbf{x}}$. To utilize the network to maximum capacity the input to the network is the ego-motion/odometry based warped event frame such that the background pixels in the two input event frames are almost aligned and the only misalignment comes from the IMOs. This ensures that the network's capacity can be utilized fully for learning sub-pixel accurate optical flow for IMO regions. The input to the EVSegFlowNet is $\mathcal{W}\left(\mathcal{E}_t, \widetilde{H}_{4Pt}\right)$ and $\mathcal{E}_{t+1}$. Here, $\widetilde{H}_{4Pt}$ is transformed to E^F before warping.

A complexity analysis of EVSegFlowNet is given in Suppl. Sec. S.VI. The success of our approach can be seen from the experimental results. The loss function for learning $\tilde{\dot{\mathbf{p}}}_{\mathbf{x}}$ is:

$$\underset{\tilde{\dot{\mathbf{p}}}_{\mathbf{x}}}{\operatorname{argmin}} \ \mathbb{E}\left(\mathcal{D}\left(\mathcal{W}\left(\mathcal{E}'_t, \tilde{\dot{\mathbf{p}}}_{\mathbf{x}}\right) \circ \mathbb{1}_f, \mathcal{E}_{t+1} \circ \mathbb{1}_f\right)\right) + \lambda_1 \mathbb{E}\left(\|\tilde{\dot{\mathbf{p}}}_{\mathbf{x}} \circ \mathbb{1}_b\|_1\right) + \lambda_2 \mathbb{E}\left(\|\tilde{\dot{\mathbf{p}}}_{\mathbf{x}} \circ \mathbb{1}_b\|_2^2\right) \quad (3)$$

Here, λ_1 and λ_2 are regularization parameters. This loss function is essentially the image difference with elastic net like regularization penalty. This penalty makes the network make background flow zero fairly quickly as compared to simple l_1 or quadratic penalty whilst being robust to outliers (errors in segmentation mask creation).

Note that all our networks were trained in simulation and directly transfer to the real world without any re-training/fine-tuning. We call our dataset *Moving Object Dataset (MOD)*. Detailed information about the dataset can be found in Suppl. Sec. S.VII.

III. Control Policy For Dodging IMOs

In this section, we present a solution for evading multiple known and/or unknown IMOs.

Let us consider three different flavors of IMOs: (i) Sphere with known radius r, (ii) Unknown shaped objects with known bound on the size and (iii) Unknown objects with no prior knowledge. We tackle each of these cases differently. Knowing the prior information about the geometric nature helps us achieve much more robust results and fine-grain control. We define $\mathcal{F}$ as the projection of all the IMOs on the image plane such that $\mathcal{F} = \bigcup_{\forall i} \mathcal{F}_i$, where $\mathcal{F}_i$ denotes the i^{th} IMO's image plane projection. Now, let's discuss each flavor of the problem separately in the following subsections.

A. Sphere with known radius r

Let us first begin with the simplest case, i.e., *a single spherical IMO with known radius r*. Evading such an object under no gravitational influence has been tackled and well analyzed by [1]. It is known that the projection of a sphere on the image plane is an ellipse [28]. For spherical objects under the gravitational influence, we estimate the initial 3D position using the known radius information and then we track the object over a few $\mathcal{E}$ to obtain the initial 3D velocity. Here, the tracking is done by segmentation on every frame.

Assuming a classical physics model, we predict the future trajectory $\mathbf{X}_i^{\text{IMO}}$ of the sphere when it is only under the influence of gravity. Now, we define the point $\mathbf{X}_{i,p}^{\text{IMO}}$ as the intersection of the trajectory $\mathbf{X}_i^{\text{IMO}}$ and the image plane. For the case of a single spherical IMO, we compute the distance between $\mathbf{X}_{i,p}^{\text{IMO}}$ and the initial position of the quadrotor O, denoted by vector $\mathbf{x}_{\text{min}} \in \mathbb{R}^{2\times 1}$. The "safe" direction is represented as $\mathbf{x}_s = -\mathbf{x}_{\text{min}}$. A simple Proportional-Integral-Derivative (PID) controller based on the differential flatness model of the quadrotor is used with high proportional gain for a quick response to move in the direction $\mathbf{x}_s$. The minimum amount of movement is equal to the extended obstacle size (the size of the quadrotor is added to the object size).

Now, let's extend to the evasion of *multiple spherical IMOs*. We assume that while objects are detected, there is no occlusion among different IMOs in the front event camera frame. Then, each object $\mathcal{F}_i$ is clustered using mean shift clustering. For each object $\mathcal{F}_i$, the 3D position and velocity are estimated as before. It is important to note that since all the objects were targeted at the quadrotor, they are bound to intercept the image plane, say at point $\mathbf{X}_{i,p}^{\text{IMO}}$ (Fig. 4). For evasion from multiple objects, we adapt the following approach. First, we find the two objects m and $m+1$ from a consecutive cyclic pair of vectors such that (here $\hat{(\cdot)}$ represents a unit vector):

$$\underset{\mathbf{X}_{i,p}^{\text{IMO}}, \mathbf{X}_{i+1,p}^{\text{IMO}}}{\operatorname{argmin}} \left\langle \hat{\mathbf{X}}_{i,p}^{\text{IMO}}, \hat{\mathbf{X}}_{i+1,p}^{\text{IMO}} \right\rangle \quad (4)$$

In other words, the objects m and $m+1$ forms the largest angle among all the consecutive cyclic pairs. So we deploy a

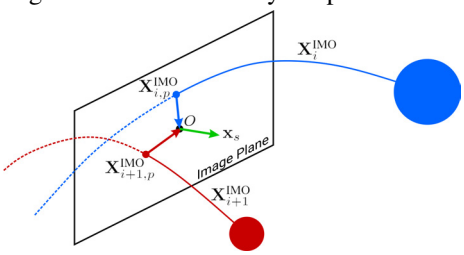

Fig. 4. Vectors $\mathbf{X}_{i,p}^{\text{IMO}}$ and $\mathbf{X}_{i+1,p}^{\text{IMO}}$ represent the intersection of the trajectory and the image plane. $\mathbf{x}_s$ is the direction of the "safe" trajectory. All the vectors are defined with respect to the center of the quadrotor projected on the image plane, O. Both of the spheres are of known radii.

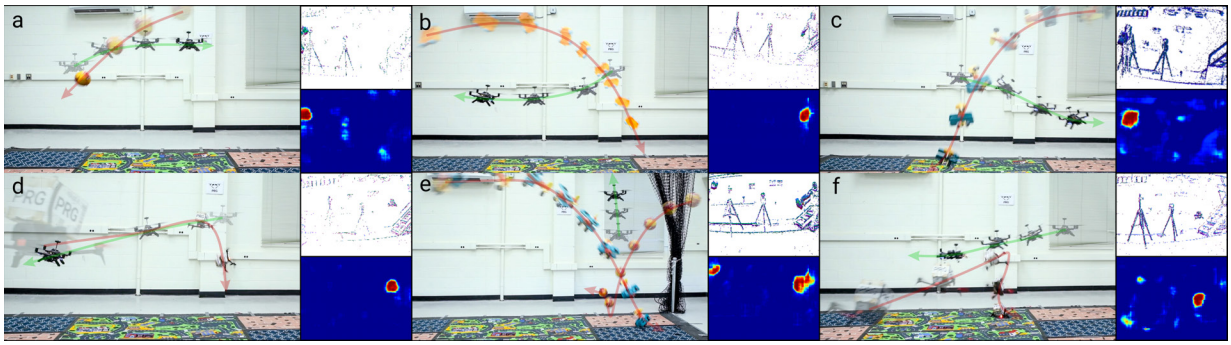

Fig. 5. Sequence of images of quadrotor dodging or pursuing of objects. (a)-(d): Dodging a spherical ball, car, airplane and Bebop 2 respectively. (e): Dodging multiple objects simultaneously. (f): Pursuit of Bebop 2 by reversing control policy. Object and quadrotor transparency show progression of time. Red and green arrows indicate object and quadrotor directions respectively. On-set images show front facing event frame (top) and respective segmentation obtained from our network (down).

strategy to move the quadrotor in $\mathbf{x}_s$ direction in the image plane such that

$$\mathbf{x}_s = \begin{cases} -\hat{\mathbf{X}}_\beta, & \text{if } \max_{\forall i}\langle \hat{\mathbf{X}}_\beta, \mathbf{X}_i\rangle < \max_{\forall i}\langle -\hat{\mathbf{X}}_\beta, \hat{\mathbf{X}}_i\rangle \\ \hat{\mathbf{X}}_\beta, & \text{otherwise} \end{cases} \tag{5}$$

where $\mathbf{X}_\beta = \hat{\mathbf{X}}^{\text{IMO}}_{m,p} + \hat{\mathbf{X}}^{\text{IMO}}_{m+1,p}$

For unknown shaped objects with bound on size, please refer Suppl. Sec. S.VIII.

B. Unknown objects with no prior knowledge

Without any prior knowledge about the object, it is geometrically infeasible to obtain the 3D velocity of an IMO using a monocular camera. However, we can predict a possible safe trajectory $\mathbf{x}_s$ depending on the velocity direction of the IMOs on the image plane. We compute the unit vector $\mathbf{v}_i^{\text{IMO}}$ in which the IMO is moving by tracking the segmentation mask of the IMO or by computing the mean optical flow direction of the region of interest. For a single unknown IMO, a heuristic is chosen such that the quadrotor moves in the direction perpendicular to the velocity of the IMO on the image plane, i.e., a safe direction for the quadrotor motion which satisfies $\langle \mathbf{x}_s, \mathbf{v}_i^{\text{IMO}}\rangle = 0$.

For evasion from multiple objects, we adapt a similar approach as in Sec. III-A. First, we find the two objects m and $m+1$ from a consecutive cyclic pair of velocity vectors by replacing $\hat{\mathbf{X}}$ by $\hat{\mathbf{v}}$ in Eq. 4. Now. we deploy a strategy to move the quadrotor in $\mathbf{x}_s$ direction in the image plane by replacing $\hat{\mathbf{X}}$ by $\hat{\mathbf{v}}$ and '$<$' by '$>$' in Eq. 5. Refer to Suppl. Sec. S.IX for an extension of our work to pursuit.

IV. Experiments

A detailed description of the hardware and experimental setup is given in Suppl. Sec. S.X.

A. Experimental Results and Discussion

In this paper, we considered the case of navigating through different sets of multiple dynamic obstacles. We dealt with six different evading combinations and one pursuit experiment: (a) Spherical ball with a known radius of 140 mm, (b) car with a bound on the maximum dimension size of 240 mm (with maximum error of $\sim 20\%$ from the original size), (c) airplane with no prior information, (d) Bebop 2 flying at a constant velocity, (e) multiple unknown objects, (f) pursuit of Bebop 2 and (g) low-light dodging experiment. For each evasion case, the objects (Suppl. Fig. S.8) are directly thrown towards the Aero quadrotor such that a collision would definitely occur if the Aero holds its initial position. For each case, a total of 30 trials were conducted. Notice that the objects would have hit the quadrotor if it had not moved from its initial position. We achieved a remarkable success rate of 86% in cases (a) and (b), 76% in case (c). Both Parrot Bebop 2 experiments (case (d), (f)) resulted in 83% success rate. Case (e) was carefully performed with synchronized throws between the two objects and resulted about 76% success rate. For the low-light experiment (case (g)), we achieved a success rate of 70%. Here success is defined as both a successful detection and evasion for the evade experiments and both a successful detection and collision for the pursuit task. Fig. 5 shows sequences of images for cases (a)-(f) along with sample front facing event frame and segmentation outputs. Vicon plots can be found in Suppl. Fig. S.9.

Before the IMO is thrown at the quadrotor, the quadrotor maintains its position (hover) using the differential X^W and Y^W estimates from the EVHomographyNet and Z^W estimates from the sonar.

When the IMO is thrown at the quadrotor, the IMO is detected for five consecutive frames to estimate either the trajectory or image plane velocity and to remove outliers using simple morphological operations. This gives a perception response lag of 60 ms (each consecutive frame pair takes 10 ms for the neural network computation and 2 ms for the post-processing). Finally, the quadrotor moves using the simple PID controller presented before.

Note that, we talked about obtaining both segmentation and optical flow from EVSegFlowNet. This was based on the conceptualization of optical flow being used for other tasks as well. However, if only the dodging task is to be performed, a smaller segmentation network can be used without much loss of accuracy.

Fig. 6 shows the input and output of EVDeBlurNet for losses $\mathcal{D}_2$ and $\mathcal{D}_3$ under $\delta t = \{1, 5, 10\}$ ms. Observe the amount of noise (stray events not associated with strong contours) in the raw images (top row of Fig. 6). The second row shows the output of EVDeBlurNet for $\mathcal{D}_2$ loss. Observe that this works well for smaller integration times but for larger integration times, the amount of denoising and deblurring performance deteriorates. However, $\mathcal{D}_3$ loss which is aimed at outlier rejection is more suppressive of weak contours and hence one can observe that the frame has almost no output for smaller integration times. This has the effect of working well for larger integration times.

Fig. 7 shows the output of EVHomographyNet using the

TABLE I

QUANTITATIVE EVALUATION OF DIFFERENT METHODS FOR HOMOGRAPHY ESTIMATION.

Method (Loss)	RMSE_i in px.					RMSE_o in px.				
	$\gamma = \pm[0,5]$	$\gamma = \pm[6,10]$	$\gamma = \pm[11,15]$	$\gamma = \pm[16,20]$	$\gamma = \pm[21,25]$	$\gamma = \pm[0,5]$	$\gamma = \pm[6,10]$	$\gamma = \pm[11,15]$	$\gamma = \pm[16,20]$	$\gamma = \pm[21,25]$
Identity	3.92 ± 0.83	11.40 ± 0.70	18.43 ± 0.70	25.50 ± 0.70	32.55 ± 0.71	3.92 ± 0.84	11.40 ± 0.70	18.44 ± 0.71	25.49 ± 0.70	32.55 ± 0.71
S	3.23 ± 1.13	3.90 ± 1.34	5.31 ± 2.05	9.63 ± 4.57	17.65 ± 7.00	4.15 ± 1.78	5.05 ± 2.19	6.99 ± 3.11	11.21 ± 4.84	18.37 ± 6.61
US* ($\mathcal{D}_1$)	2.97 ± 1.22	3.84 ± 1.61	5.99 ± 2.78	11.64 ± 5.69	20.36 ± 7.68	3.92 ± 1.53	5.31 ± 2.43	8.14 ± 3.86	13.63 ± 5.87	21.22 ± 7.35
US* ($\mathcal{D}_2$)	2.48 ± 0.93	3.53 ± 1.43	5.89 ± 2.70	11.74 ± 5.69	20.51 ± 0.70	3.19 ± 1.26	4.86 ± 2.31	7.92 ± 3.73	13.47 ± 5.71	21.22 ± 7.08
DB + S	2.73 ± 1.01	3.16 ± 1.23	**4.00 ± 1.79**	**6.50 ± 3.54**	**12.22 ± 6.58**	3.69 ± 1.51	**4.49 ± 2.10**	**5.91 ± 3.16**	**9.04 ± 4.90**	**14.60 ± 6.95**
DB + US ($\mathcal{D}_1$)	**2.19 ± 0.88**	**3.04 ± 1.57**	4.99 ± 2.75	10.16 ± 5.54	18.62 ± 7.85	**3.08 ± 1.37**	4.63 ± 2.68	7.57 ± 4.30	13.16 ± 6.25	21.08 ± 7.49
DB + US ($\mathcal{D}_2$)	2.41 ± 1.06	3.30 ± 1.77	5.36 ± 3.02	10.39 ± 5.78	18.77 ± 8.07	3.35 ± 1.76	5.05 ± 3.03	8.11 ± 4.65	13.46 ± 6.48	21.08 ± 7.81

* Trained for 100 epochs on supervised and then fine-tuned on unsupervised for 100 more epochs. γ denotes the perturbation range in px. for evaluation.

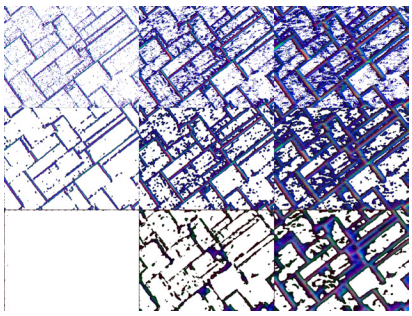

Fig. 6. Output of EVDeBlurNet for different integration time and loss functions. Top row: raw event frames, middle row: deblurred event frames with $\mathcal{D}_2$ and bottom row: deblurred event frames with $\mathcal{D}_3$ with δt. Left to right: δt of 1 ms, 5 ms and 10 ms. Notice that only the major contours are preserved and blurred contours are thinned in deblurred outputs.

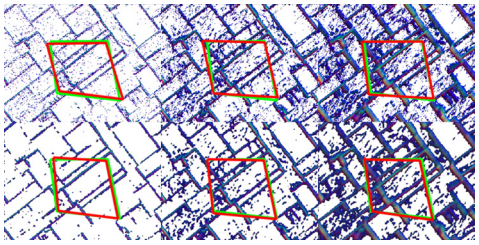

Fig. 7. Output of EVHomographyNet for raw and deblurred event frames at different integration times. Green and red color denotes ground truth and predicted $\widetilde{H}_{4Pt}$ respectively. Top row: raw events frames and bottom row: deblurred event frames. Left to right: δt of 1 ms, 5 ms and 10 ms. Notice that the deblurred homography outputs are almost not affected by δt.

supervised loss function on both raw (top row) and deblurred frames (bottom row). Observe that the deblurred homography estimation is more robust to changes in different integration times. The extended version of Table I is available in Suppl. Sec. S.XI.) shows the quantitative evaluation of different methods used for training EVHomographyNet. Here, DB represents deblurring using the combination of $\mathcal{D}_2$ and $\mathcal{C}_2$ loss, S and US refer to supervised and unsupervised losses respectively. RMSE_i and RMSE_o denote the average root mean square error [25] in the testing dataset with textures similar to that of the training set, and completely novel textures respectively. RMSE_o quantifies how well the network can generalize to unseen samples. Notice that the supervised flavors of the algorithm work better (lower RMSE_i and RMSE_o) than their respective unsupervised counterparts. We speculate that this is because of the sparsity in data and that the simple image based similarity metrics not being well suited for event frames. We leave crafting a novel loss function for event frames as a potential avenue for future work. Also, notice how deblur variants of the algorithms almost always work better than their respective non-deblurred counterparts highlighting the utility of EVDeblurNet.

Table II shows the quantitative results of different variants of segmentation networks trained using the $\mathcal{D}_2$ loss for

TABLE II

QUANTITATIVE EVALUATION OF IMO SEGMENTATION METHODS.

Method (Loss)	DR_i in %	DR_o in %	Run Time in ms	FLOPs in M	Num. Params in M
SegNet	49.0	40.4	1.5	222	0.03
DB + SegNet	81.5	68.7	7.5	4900	2.33
DB + H + SegNet	83.2	69.1	10	5150	3.63
SegFlowNet	88.3	81.9	1.5	222	0.03
DB + SegFlowNet	93.3	90.1	7.5	4900	2.33
DB + H + SegFlowNet	93.4	90.7	10	5150	3.63

SegFlowNet. Also, H denotes the stack of warped $\mathcal{E}_t$ and $\mathcal{E}_{t+1}$ using the outputs of the network DB + S in Table I. Here DR denotes the detection rate and is defined as:

$$\text{DR} = \mathbb{E}\left(\overline{(\mathcal{D} \cap \mathcal{G}) \circ \mathbb{1}_{\mathcal{E}}} / \overline{(\mathcal{G} \circ \mathbb{1}_{\mathcal{E}})} \geq \tau\right) \times 100\% \tag{6}$$

where $\mathcal{G}$ and $\mathcal{D}$ denote the ground truth and predicted masks respectively, and $\mathbb{1}_{\mathcal{E}}$ denotes the presence of an event in either of the input event frames. For our evaluation, we choose $\tau = 0.5$. Notice that using both deblur and homography warping helps improve the results as anticipated. Again, DR_i and DR_o denote testing on trained objects and completely novel objects. As before, deblurring helps generalize much better to novel objects and deblurring with homography warping gives better results showing that the network's learning capacity is utilized better. Also, notice that the improvement in segmentation by warping using homography (last row) is marginal due to the 3D structure of the scene. The network architectures and training details are provided in Suppl. Sec. S.V.

V. CONCLUSIONS

We presented an AI-based algorithmic design for micro/nano quadrotors, taking into account the knowledge of the system's computation and latency requirements using deep learning. The central conception of our approach is to contrive AI building blocks using shallow neural networks which can be re-purposed. This philosophy was used to develop a method to dodge dynamic obstacles using only a monocular event camera and on-board sensing. To our knowledge, this is the first deep learning based solution to the problem of dynamic obstacle avoidance using event cameras on a quadrotor. Moreover, our networks are trained in simulation and directly transfer to the real world without fine-tuning or retraining. We also show the generalizability of our navigation stack by extending our work to the pursuit task. As a parting thought, a better similarity scoring metric between event frames or a more robust construction of event frames can improve our results.

ACKNOWLEDGEMENT

This work was partly funded by the Brin Family Foundation, National Science Foundation under grant BCS 1824198, ONR under grant N00014-17-1-2622, the Northrop Grumman Mission Systems University Research Program. The authors would like to thank NVIDIA for the grant of an Titan-Xp GPU and Intel for the grant of the Aero Platform.

REFERENCES

[1] D. Falanga et al. How Fast Is Too Fast? The Role of Perception Latency in High-Speed Sense and Avoid. *IEEE Robotics and Automation Letters*, 4(2):1884–1891, April 2019.

[2] P. Sermanet et al. Speed-range dilemmas for vision-based navigation in unstructured terrain. *IFAC Proceedings Volumes*, 40(15):300–305, 2007.

[3] G. Gallego, , et al. Event-based vision: A survey. *arXiv preprint arXiv:1904.08405*, 2019.

[4] A. Vidal et al. Ultimate SLAM? Combining events, images, and IMU for robust visual SLAM in HDR and high-speed scenarios. *IEEE Robotics and Automation Letters*, 3(2):994–1001, 2018.

[5] M. Bloesch et al. Robust visual inertial odometry using a direct ekf-based approach. In *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 298–304. IEEE, 2015.

[6] T. Qin et al. VINS-Mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, 34(4):1004–1020, 2018.

[7] A. Zhu et al. Event-based visual inertial odometry. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5816–5824. IEEE, 2017.

[8] H. Liang et al. SalientDSO: Bringing attention to direct sparse odometry. *IEEE Transactions on Automation Science and Engineering*, 2019.

[9] S. Bowman et al. Probabilistic data association for semantic SLAM. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1722–1729. IEEE, 2017.

[10] R. Sabzevari and D. Scaramuzza. Multi-body motion estimation from monocular vehicle-mounted cameras. *IEEE Transactions on Robotics*, 2016.

[11] V. Vasco et al. Independent motion detection with event-driven cameras. In *2017 18th International Conference on Advanced Robotics (ICAR)*, pages 530–536. IEEE, 2017.

[12] A. Mitrokhin et al. Event-based moving object detection and tracking. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–9, Oct 2018.

[13] T. Stoffregen et al. Event-based motion segmentation by motion compensation. In *International Conference on Computer Vision (ICCV)*, 2019.

[14] A. Mitrokhin et al. Ev-imo: Motion segmentation dataset and learning pipeline for event cameras. *arXiv preprint arXiv:1903.07520*, 2019.

[15] H. Alvarez et al. Collision avoidance for quadrotors with a monocular camera. In *Experimental Robotics*, pages 195–209. Springer, 2016.

[16] N. Sanket et al. GapFlyt: Active vision based minimalist structure-less gap detection for quadrotor flight. *IEEE Robotics and Automation Letters*, 3(4):2799–2806, Oct 2018.

[17] A. Barry et al. High-speed autonomous obstacle avoidance with pushbroom stereo. *Journal of Field Robotics*, 35(1):52–68, 2018.

[18] K. Mohta, , et al. Fast, autonomous flight in gps-denied and cluttered environments. *Journal of Field Robotics*, 35(1):101–120, 2018.

[19] K. Mohta et al. Experiments in fast, autonomous, gps-denied quadrotor flight. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7832–7839. IEEE, 2018.

[20] E. Mueggler et al. Towards evasive maneuvers with quadrotors using dynamic vision sensors. In *2015 European Conference on Mobile Robots (ECMR)*, pages 1–8. IEEE, 2015.

[21] G. Gallego et al. A unifying contrast maximization framework for event cameras, with applications to motion, depth, and optical flow estimation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[22] G. Gallego et al. Event-based camera pose tracking using a generative event model. *arXiv preprint arXiv:1510.01972*, 2015.

[23] A. Zhu et al. Unsupervised event-based learning of optical flow, depth, and egomotion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 989–997, 2019.

[24] D. DeTone et al. Method and system for performing convolutional image transformation estimation, November 23 2017. US Patent App. 15/600,545.

[25] Ty Nguyen et al. Unsupervised deep homography: A fast and robust homography estimation model. *IEEE Robotics and Automation Letters*, 3(3):2346–2353, 2018.

[26] S. Meister et al. Unflow: Unsupervised learning of optical flow with a bidirectional census loss. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[27] A. Zhu et al. Ev-flownet: self-supervised optical flow estimation for event-based cameras. *arXiv preprint arXiv:1802.06898*, 2018.

[28] C. Wylie. *Introduction to projective geometry*. Courier Corporation, 2011.