Preprint version of:
Pérez-Cutiño, M.A., Gómez-Eguíluz, A., Martínez-de Dios, J.R. and Ollero, A. (2021, April).
Event-based human intrusion detection in UAS using Deep Learning.
In *IEEE International Conference on Unmanned Aircraft Systems (ICUAS 2021).*

Event-based human intrusion detection in UAS using Deep Learning

M.A. Pérez-Cutiño, A. Gómez Eguíluz, J.R. Martínez-de Dios and A. Ollero

***Abstract*— Automatic intrusion detection in unstructured and complex environments using autonomous Unmanned Aerial Systems (UAS) poses perception challenges in which traditional techniques are severely constrained. Event cameras have high temporal resolution and dynamic range, which make them robust against motion blur and lighting conditions. This paper presents an event-by-event processing scheme for detecting human intrusion using UAS. It includes: 1) one method for detecting clusters of events caused by moving objects in static background; and 2) one method based on Convolutional Neural Networks to compute the probability that a cluster corresponds to a person. The proposed scheme has been implemented and validated in challenging scenarios.**

***Index Terms*— event camera, surveillance, aerial robots, deep learning.**

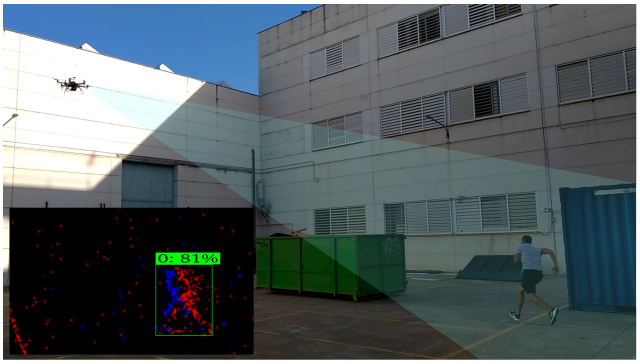


Fig. 1: Experimentation of event-based human intrusion detection using aerial robots.

I. Introduction

Intrusion detection using autonomous Unmanned Aerial Systems (UAS) in unstructured and complex environments poses very challenging perception problems. The operation of traditional sensors such as cameras and LIDARS are usually constrained by the lack of structure in the scenario, lighting conditions, or camera motion blur due to the robot vibrations or movements. The advent of event cameras opens new sensing possibilities of interest for aerial robot surveillance. Event cameras have high temporal resolution, and hence are not affected by motion blur. They have high robustness to lighting conditions, being able to operate during day and night, avoiding the need to install two cameras as in traditional day/night configurations, which affect the UAS payload, electrical consumption, and computational requirements. Besides, event cameras have moderate weight and low power consumption, which make them suitable for installation on board of small aerial robots.

In previous works [1] we presented an event-based processing scheme that detected intruders using aerial robots. The method is capable of detecting moving objects, but cannot determine the type of object that originated the detection.

This paper presents a scheme for event-based human intrusion detection for UAS. It is based on two main event-based processing modules. The first module detects intrusions as clusters of events with consistent motion. The second asynchronously analyzes the events of each of the detected clusters (events not associated to a detected cluster are not processed to save computational cost), and computes the probability that the cluster corresponds to a person. Each event corresponding to a detected cluster is processed using Deep Learning (DL) techniques (specifically, Convolutional Neural Networks (CNN)) to compute the probability that the event corresponds to a person. The computed event probabilities are used to estimate the probability that a cluster corresponds to a person. The main novelty of our method resides on two facts. First, it is based purely on events removing the need for other sensors on the UAS and resulting in high robustness against motion blur and lighting conditions. Second, the events are event-by-event processed by both modules, fully exploiting the asynchronous capabilities of event cameras. The proposed method was implemented in an hexarotor and experimentally validated in challenging scenarios, see Figure 1.

The rest of the paper is structured as follows. The main related work is briefly summarized in Section II. Section III presents the general scheme of the event-based aerial robot surveillance system. Section IV presents the proposed Deep Learning method for event-based people detection. The validation experiments in challenging scenarios are presented in Section V. The conclusions are summarized in Section VI.

II. Related Work

The use of aerial robots for surveillance tasks have attracted high R&D interest. Many techniques and systems have been developed focusing on perception, planning, or multi-robot coordination, among many others. One of the most relevant functionalities in these systems is the automatic detection and tracking of intruders, which is often approached as the detection of moving objects. Many techniques based on the processing of visual images for automatic pedestrian detection, face recognition, motion seg-

This work was supported by the European Research Council as part of GRIFFIN ERC Advanced Grant 2017, Action 788247 and ARM-EXTEND (DPI2017-8979-R) project funded by the Spanish National R&D Plan.
The authors are with the GRVC Robotics laboratory, University of Seville, Seville 41092, Spain email: {mpcutino, jdedios, ageguiluz, aollero}@us.es

mentation, and target tracking, have been proposed. However, the motion blur and lighting conditions sensitivity of traditional visual cameras constrain their efficacy in unstructured outdoor scenarios.

The advantages of event cameras have motivated their use in robotics and computer vision [2]. Event cameras provide high dynamic range, low latency, low energy consumption, and robustness to motion blur. A number of works have explored human detection using event cameras. Most of them rely on processing *event images* that accumulate the events either by time or a fixed number of samples. The work in [3] compared the performance of traditional and event cameras for human faces detection, showing the viability of using only events for face detection. Another work with surveillance applications was presented in [4]. The output of two YOLO object detectors were fused to detect pedestrians using standard camera frames and *event images* from a fixed-position DAVIS camera. Other works such as [5] and [6] used *event images* to segment the scene through the motion of the objects. Despite all of them providing interesting solutions to surveillance related problems using event cameras, their evaluation on real robots was not approached.

Recently, the use of event cameras for UAS perception have been explored by the robotics community. A motion segmentation method was implemented in [7] in order to allow a multirotor UAS to perform collision avoidance using an stereo set-up of event cameras, which was used to track the moving objects thrown at it. The work in [8] presented another method for tracking moving objects using *event images* and compensating for the global motion of a Micro Aerial Vehicle (MAV).

All the above methods use *event images* to transform group of events into image-like representations and, therefore, they do not fully exploit asynchronous, sparse, and sequential nature of event data. In contrast, asynchronous *event-by-event* processing methods consider the intrinsic nature of event data. A number of works have explored *event-by-event* processing for feature detection and tracking [9] [10], clustering [11] [1], or localization [12]. However, few of them have explored the application of event-by-event processing on real robots as the computational requirements are often too high to be computed on board.

A number of event-based methods for object classification methods have been developed. Inspired by the success of object detectors based on traditional visual images such as [13]–[15], the first approach in object detection using events relied on processing *event images*. Some works for object detection and classification using *event images* have already been mentioned [3], [4]. Others focus on novel image-like representations to provide robust classification performance [16]–[18]. In [16] they modify the local spatial descriptors defined in [19] to take into account events occurring on the same pixel, and then accumulate the histograms of events in predefined zones of the image. Work [17] defined Event Spike Tensors through kernel convolutions, quantizations, and projections, and work [18] proposed the use of an adaptive motion-agnostic encoder to integrate event signals into frame-based representations. Most of the aforementioned works relies on Support Vector Machines (SVM) and/or CNN to perform the classification task. In [20] a hardware implementation of a Binary Neural Network (BNN) is used for pedestrian detection. The results were compared with a standard CNN which overcome the BNN in terms of accuracy. Despite of the promising results on these works, they have not been tested on real robots.

Some efforts have been made to introduce event-based object detection methods on ground robots. A CNN was trained in [21] using *event images* and grayscale images to guide a non-holonomic Unmanned Ground Vehicle in a predator-prey experiment. An iCub robot was presented in [22] where *event images* were used for object detection using an off-the-shelf Deep Learning approach. However, motion and location of aerial vehicles produce different event streams than on ground vehicles. In the recent work of [23] the first DL approach to dodge incoming objects was proposed. They use a framework with three neural networks combined for different tasks. However, they do not deal with the classification of the detected object.

Event data is asynchronous and sparse. Merging asynchronous and sparse data for their use as input of artificial neural networks field, is a complex task. The works of [24], [25] transformed traditional convolutional networks in sparse structures to process events asynchronously. In [24] they proposed the integration of YOLO and a leaky surface to reformulate recursive update rules for convolution and pooling operations. In [25] they define a rulebook to perform Submanifold Sparse Convolution that updates the network prediction according to new active or inactive sites. However, these techniques are limited in the types of representations that can be processed [24] or require very high computational burden taking more than $80ms$ for the processing of a single event [25]. Another strategy arises on the use of Spiking Neural Networks (SNN). Although SNN process data asynchronously, they are hard to train because of the lack of a proper update rule in the learning stage. For this reason, many works adopt first train a standard convolutional networks and then transform them to SNN [26], [27]. Moreover, results using CNN usually surpass their SNN counterpart.

Event-by-event processing for object detection and classification on real aerial robots remains an understudied field. A Field-Programmable Gate Array (FPGA) is used in [28] to classify and detect objects based on events counts, Principal Component Analysis (PCA) and SVM. The results obtained surpass state-of-the-art methods for object classification in benchmark datasets. However, the validation was made in indoor scenarios where all objects were static and the drone movement was slow. Moreover, specialized hardware is required to replicate their work.

III. Event-based target detection and tracking

A wide variety of sensors for aerial robot surveillance have been used [29]. Traditional frame-based cameras are small, lightweight, inexpensive, and provide texture and

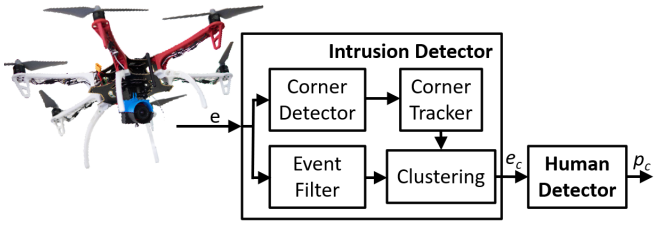


Fig. 2: Architecture of the proposed scheme.

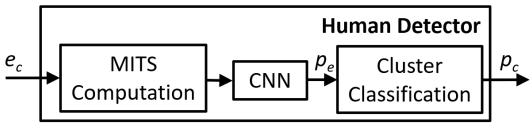


Fig. 3: Scheme of *Human Detector*.

color, which are very valuable for object detection and identification. However, they are sensitive to lighting conditions and can suffer from motion blur, which severely constrain their use in surveillance applications. Infrared cameras are interesting in cases where the targets have different temperature from the background. In the last years strong reductions in their size, weight, and cost have motivated their popularization in civilian surveillance applications [30]. However, the low sensitivity of infrared camera detectors result in large exposure times, frequently causing motion blur when operating on board aerial robots. Range sensors to obtain distance measurements between the robot and radio tags deployed in the scenario or installed on objects or other robots, have been used for aerial robot navigation, see e.g. [31], and some application-related perception tasks, such as object search [32]. However, the radio tag should be transported by the intruder, which is not a realistic assumption in surveillance applications.

This work is based on event cameras, which capture the visual information in the form of events. Events are triggered asynchronously with high temporal resolution (μs), and represent changes of illumination in the scene. Event cameras have good properties for aerial robot perception in unstructured scenarios. They can cope with robot motion and vibrations due to their very low latency without motion blur [10]. Their high dynamic range (> 100 dB) provide robustness against changes in lighting conditions. Finally, they have moderate size and weight, and low energy consumption.

Our surveillance system processes the events gathered by an event camera on board a UAS to detect and identify people. The surveillance scenarios are assumed static except for the targets. This is the case of many scenarios in industrial, building, or home security surveillance applications. Any object/person/animal in motion creates groups of events close to corners with globally consistent motion. However, considering all of them as intruders would result in a high number of false alarms. Hence, a module is necessary to confirm if the detection is actually originated by a person.

The presented event-based processing scheme for UAS-based surveillance includes two main systems, see Figure 2. First, *Intrusion Detector* processes the event stream to detect intruders represented as clusters of events close to corners with globally consistent motion in the scenario. It outputs the events corresponding to the detected clusters. Second, *Human detector* analyzes the events of the detected clusters to determine the probability that they correspond to a person.

Intrusion Detector is based on the detection and tracking system we proposed in [1]. It includes several modules that process the event stream received from the event camera. On board aerial robots, triggered events can be caused by moving objects (potential intruders) or by static background with robot motion. First, an event filtering module differentiates between both cases relying on the event spatio-temporal information to capture the regions that trigger more events within a time frame. The events falling out of these regions are assigned as originated by static background and are filtered out. Next, an event-based corner detection method is applied. We selected *eFast [9] due to its trade-off between accuracy, performance, and computational efficiency. Next, the detected corners are tracked to remove inconsistencies and noise. Intruders create groups of events close to corners with consistent motion. The corners are useful to detect objects but do not have sufficient information. Clusters with close events and corners are created using criteria based on distance on the image plane. Finally, the obtained clusters represented by their centroid are tracked. Clusters are joined if they satisfy distance and event-density criteria. If a cluster is not updated consistently, it is filtered out. For further details of the asynchronous event processing algorithms implemented within *Intrusion Detector* refer to our previous works [1] and [33].

IV. Deep Learning Method for Event-Based People Identification

Intrusion Detector detects clusters with corners and events caused by moving objects that have consistent motion in the scene, and outputs the events belonging to these clusters. However, these clusters can correspond to objects, animals, or people. *Human Detector*, Figure 3, analyzes the events belonging to each detected cluster and determines the probability that it corresponds to a person. The clusters detected by *Intrusion Detector* represent Regions of Interest (ROI). Every event falling in a ROI is asynchronously processed. First, a CNN analyzes a spatio-temporal descriptor of each event e_c and estimates the probability p_e that e_c corresponds to a person. Second, the probability p_c that the cluster corresponds to a person is computed using the probability values from all its events.

Our event classification method is based on Convolutional Neural Networks. CNNs are bioinspired regularized versions of multilayer perceptrons in which the connectivity pattern between neurons resembles the organization of the animal visual cortex. They are maybe the type of deep neural networks most widely used in vision applications, see e.g. [13]–[15], [34], [35], and also in event-based perception [22], [36]–[38]. Using CNNs for event classification requires a dense representation of events, which generally are temporally and

spatially sparse. Hence, the first stage is building a dense representation of events.

Several event dense representations have been reported, such as [39], which groups events by polarity based on a distance metric, or [40], which maps an event count matrix in Cartesian coordinates to log polar coordinates. Work [39] is too computationally intensive to be considered in an on-time event-by-event processing while [40] does not take into account the temporal information of the events. The presented work uses time surfaces, which are spatio-temporal representations of the events that have been triggered in a region in a specified time [19]. Time surfaces can be computed very efficiently and can also be used as local descriptors of the events, and hence suitable for classification of single events as in our problem. We adopt an approach similar to [16], which proposed the use of all events occurrences in a pixel during a time interval. However, instead of using a fixed time window, our time surfaces, called *Motion Independent Time Surfaces* (*MITS*), are generated accumulating the last n events (without taking into account the event polarity) to keep a representation that is not dependent of the robot/camera velocity. For instance, with fast camera motions the resulting time surface contains the events triggered in smaller time windows than with slow camera motions.

First, the most recent events received are stored in buffer **B** of size n, which is updated with each new event following a FIFO (First In First Out) policy. Every incoming event e_i at coordinates (u_i, v_i) generates a *MITS* descriptor, which is a square local window of size MxM pixels, centered at (u_i, v_i) computed from the contributions of the events in **B**. The value of the *MITS* descriptor for event e_i at coordinate (u, v) within the *MITS* window ($u, v \in [1, M]$), is computed as follows:

$$MITS_{e_i}(u,v) = \sum_{\forall e_j \in B(u,v)} e^{-\frac{t_i - t_j}{\tau}}, \qquad (1)$$

where e_j refers to any of the events in **B** that occurred at coordinates $(u_i - M/2 + u, v_i - M/2 + v)$, t_i and t_j are the timestamps of events e_i and e_j respectively, and τ is a time constant that extends the contribution of past events in the exponential decay kernel also used in [19].

MITS are represented as gray level images of size MxM: their values are scaled between 0 and 255, and rounded to the closest integer. Hence, the *MITS* for event e_i with timestamp t_i is a gray level MxM image with the spatio-temporal representation at time t_i of the events triggered at the coordinates of event e_i. Events generated by noise will produce *MITS* descriptors with low values (close to 0). Object contours will generate events with spatial and temporal consistency and will produce *MITS* descriptors with high values (close to 255). Besides, the *MITS* of event e_i reflects the shape of the object contour that generated the event, and hence can be used to classify it. The filtering effect of *MITS* and its capability to represent objects is illustrated in the example shown in Figure 4.

MITS originated by noise or spurious effects will contain little information of the neighborhood of the event, and can lead to wrong event probability outputs. The same can occur with events triggered on low event-density regions of the scene, hence not consistent with the presence of intruders. If the *MITS* of event e_i satisfies $\sum_{u,v} MITS_{e_i}(u,v) < \eta$, e_i is not processed. This efficient filter preserves the shape of the objects in motion while removing isolated events caused by noise, involving two main positive consequences. First, it avoids processing events unlikely to be caused by intruders, saving computational burden, and favouring its onboard execution. Second, it improves the robustness of the method against noise.

Several CNN architectures were tested to find a trade-off between the number of parameters of the network and the classification accuracy. The adopted CNN is shown in Figure 5. The input is the *MITS* of the event represented as a 21x21 matrix. All layers (except for the last one) use batch normalization and dropout as regularization techniques, and the Rectified Linear Unit (ReLU) activation function; additionally, a max pooling operation was used before the first and after the last hidden layer in order to reduce the computations and to extract the sharpest features of the image [41]. Finally, in the output layer a convolutional operation is performed to obtain a 1x1 matrix, where the number of channels equals the number of classes, two in our problem. The output are the probabilities that the event analyzed corresponds to classes `human` and `non-human`. Regarding the number of channels, we first make a huge increase to allow expressiveness and then we divide it by half in each subsequent layer to reduce the network parameters. Thus, fast computation is obtained while keeping good generalization capabilities.

The CNN was trained using the Adam optimizer [41] with a loss function based on Cross-entropy but weighted to consider the different *a priori* probabilities of the classes adopted. The loss function is:

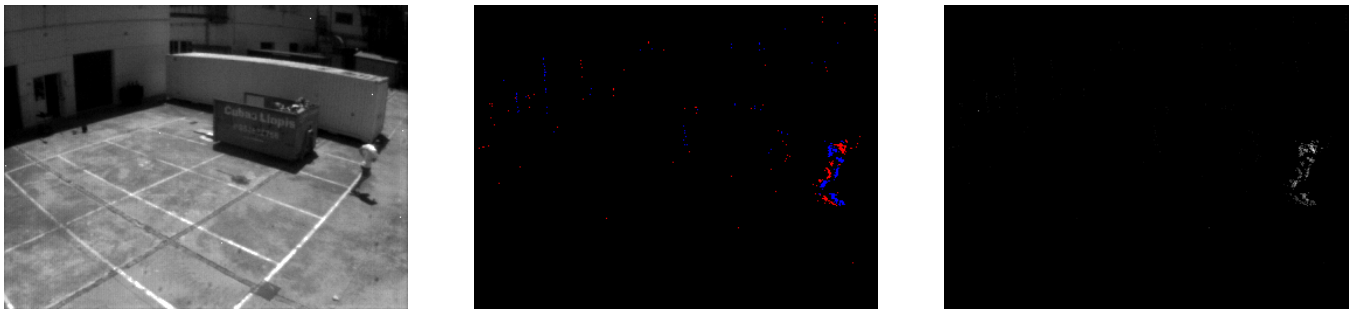
$$L(x) = -\sum_i w_i p_i(x) \log q_i(x),$$

where $p_i(x)$ is the probability of sample x of belonging to the class i, $q_i(x)$ is the probability prediction of the CNN, and w_i is a weight to take into account the different *a priori* probability of belonging to class i and is computed as:

$$w_i = \alpha_i(1 - P_i),$$

where P_i is the *a priori* probability of class i and it is estimated as the percentage of samples of class i in the training dataset, and α_i is a parameter that highlights the class relevance. w_i is useful to compensate the differences in the representation of each class and, at the same time, it allows to *push* the network prediction in a specific direction to reduce (increase) the false negatives (positives), even in case of using the same value of α_i for both classes. An analysis of the effect of α_i is shown in Section V-A.

The classification of cluster c is computed by simply averaging the probabilities of the events belonging to it, and



(a) Conventional image (b) Event image (c) Aggregation of events MITS

Fig. 4: Filtering effect and capability to preserve events at dense areas of *MITS* in examples from the experiments: a) visual conventional from the DAVIS APS sensor; b) event image from the DAVIS DVS obtained by accumulating events generated for $20ms$; and c) aggregation of the local *MITS* operators obtained in the aforementioned time interval.

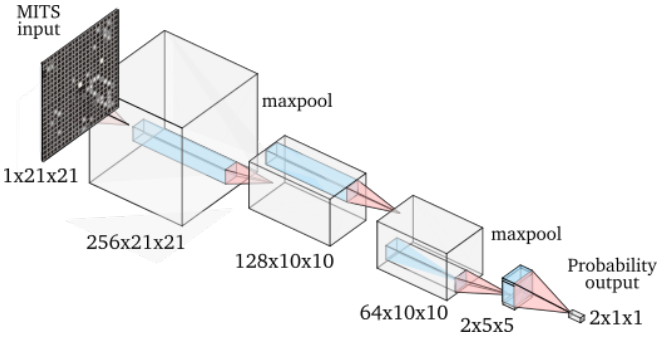


Fig. 5: Architecture of the adopted CNN for asynchronous event classification.

Fig. 6: The experimental platform: DJI Flamewheel F550 hexarotor equipped with an iniVation DAVIS346 camera.

then assigning to c the class with the highest value. This is the final output of our system.

V. Experiments

The proposed surveillance system was validated in sets of experiments in highly complex and unstructured scenarios. The UAS used was a DJI Flamewheel F550 hexarotor equipped with a DAVIS 346 and a low-cost ODROID-XU3 board for data logging, see Figure 6. The method was evaluated on an INTEL® NUC6i7KYK2, which has been used on board similar UAS platforms in intrusion monitoring applications, see e.g. [1]. The DAVIS346 embeds a 346x260 dynamic vision sensor (DVS) that outputs timestamped and polarized events and a 346x260 active pixel sensor (APS) that is coincident with the DVS and outputs grayscale images at 40 Hz. Although the proposed scheme only uses events, the visual images are also logged in the experiments for visualization. The described event-based scheme was implemented in ROS Kinetic.

Two sets of experiments were conducted. The first one validates and evaluates the *Human Detector* module, whereas the second, the full event-based surveillance system using UAS. The same CNN structure and training parameters were used in both experiments. The CNN was trained for a maximum of 40 epochs using an Adam optimizer with learning rate of 0.001 and a weight decay of 10^{-6}. The datasets used where divided in 80% for training and 20% for validation with a stratification technique to preserve the class distribution in each subset. We stop training using the Early Stop strategy, i.e., if the validation loss is not improved after a given number of epochs, 6 in our case. Besides, we save the CNN model with lower loss value in the validation set for all epochs, because theoretically it is the model with better generalization performance. To obtain the testing set metrics, new data were processed. Thus, instead of splitting in 3 parts our dataset, we test the network performance using *totally new* data. This might reduce our final metrics results, but it will show more robustness and the possibility of application to different unseen data.

The scheme performance was evaluated by *Accuracy*, *Precision*, and *Recall*, defined as:

$$Accuracy = \frac{TP + FP}{T + N}, \tag{2}$$

$$Precision = \frac{TP}{TP + FP}, \tag{3}$$

$$Recall = \frac{TP}{TP + FN}. \tag{4}$$

where T is $TP+TN$, N is $TN+FN$ and TP, TN, FP, and FN stand for the true positive, true negative, false positive, and false negative rates, respectively. As stated in [42],

using solely *Accuracy* can be misleading when evaluating algorithms that output a probability class value. Receiver Operator Characteristic (ROC) curves are not well suited for datasets with high differences on class distribution. However, performance analysis based on *Accuracy*, *Precision*, and *Recall* have been widely adopted in this kind of problems.

A. Event-based people detection evaluation

First, the capability of the *Human Detector* module to asynchronously classify events corresponding to humans is evaluated. In this experiment all the events from the event stream are used as input to *Human Detector*, i.e. without using module *Intrusion Detector* in the event processing pipeline.

For this experiment, we recorded training datasets with the APS and DVS output from the DAVIS 346 and moved the camera to generate events with different polarities. Besides humans, the scene contained objects such as ceiling lamps, windows, walls, or pillars. Events were labeled with a semi-supervised method that uses YOLO [14] fed with visual images from the DAVIS APS to detect the objects in the scene. The detections from YOLO were corrected by a human who resized, deleted, and added bounding boxes (i.e. RoIs) when required. Let B_t and B_{t+1} be the ROIs detected by YOLO as class c (class `human` in our problem) in two consecutive visual images at times t and $t+1$. All received events with timestamps between t and $t+1$ which coordinates lie within $B_t \cup B_{t+1}$ are labeled as `human`. The events out of $B_t \cup B_{t+1}$ are labeled as `non-human`. A total of 4 videos of 38 seconds mean duration where labelled using this approach, producing a total of more than 5.200 visual images and $5 * 10^6$ events. Additionally, in order to increase the number of objects in motion in the training dataset, we added data from from the *Extreme Event Dataset* (EED)[1]. The EED dataset contain multiple moving objects (people are not included) in challenging conditions.

The training datasets contain significantly more events labelled as `non-human` than, as `human`. To compensate for that difference, we adopted $\alpha = 10$ to give more weight to the classes with lower representation in the training datasets. The parameters used in the computation of *MITS* where buffer size of $|B|$=1000, *MITS* size of M=21, and time constant of τ=10^{-3}. The *MITS* filtering threshold was η=30, which removed 22% of the input events as noise.

Figure 7 shows the evolution of the loss metric of the CNN for the training and validation dataset. The *MITS* has low discrimination capabilities but the object is classified using groups of nearby events with the same classification. Hence, although the training of the CNN did not reach full convergence, the proposed system obtains good object classification, as is shown below. When the network reached the maximum number of epochs, the error curve was already in a *near-to-flat* situation.

Table I shows the performance obtained in the training, validation, and testing datasets. Training and validation

[1]http://prg.cs.umd.edu/software

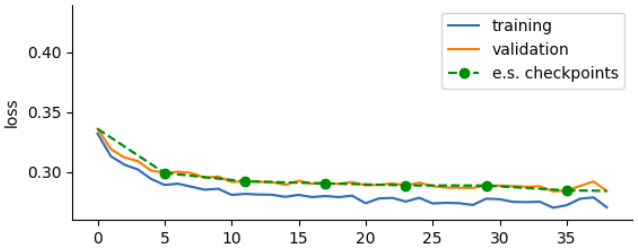


Fig. 7: Loss during CNN training as a function of the epochs, for the training and validation dataset. Green dots correspond to Early Stop checkpoints.

Dataset	Accuracy(%)	Recall(%)	Precision(%)	# of events
training	87	89	70	$\approx 3.2{*}10^6$
validation	86	88	69	$\approx 6.4{*}10^5$
testing	93	90	91	$\approx 2.4{*}10^5$

TABLE I: CNN performance in the training, validation, and testing datasets.

datasets give similar metrics results. The good values obtained for the testing dataset highlights the CNN ability to detect people. The testing dataset contained $\sim 4 * 10^4$ events corresponding to people and $\sim 2 * 10^5$ events corresponding to other objects. The adopted value α=10 succeeded in compensating for this difference.

Figure 8 shows the results of the trained CNN in several examples taken in complex scenarios different for those considered in the training: one example where the person is moving in outdoors (Figure 8-a); one example where the person is partially occluded (Figure 8-b); one example where the person is on a bike and the event camera had over-exposed lighting conditions (Figure 8-c). For visualization, the conventional visual image from the DAVIS APS in each example is shown on the left. In each example the figure on the center shows the events classified as `human` in green color. The figures on the right show the events classified as `non-human` in green color. The events of different polarity provided by the DAVIS DVS that were removed as noise by the *MITS* filter, and hence were not processed by the CNN, are shown in blue and red. Also, the regions with high density of events classified to the same class were clustered and marked with a bounding box, and the probability of the event clusters were computed and marked in the images.

It can be seen in Figure 8 that the *MITS* filter removed a significant percentage of the events caused by noise, improving robustness and reducing computational effort. Also, the proposed CNN provided remarkable classification results for `human` and `non-human` even with adverse lighting conditions. The probability values provided by the CNN were also satisfactory. Only in some occasional cases the CNN made wrong classifications, see e.g. Figure 8-c, where two persons were detected and only one was present. Some of these errors were originated by performing event clustering after event classification by the CNN. They can be solved in schemes where events are clustered before classification. This is the case when the CNN is used within the UAS-

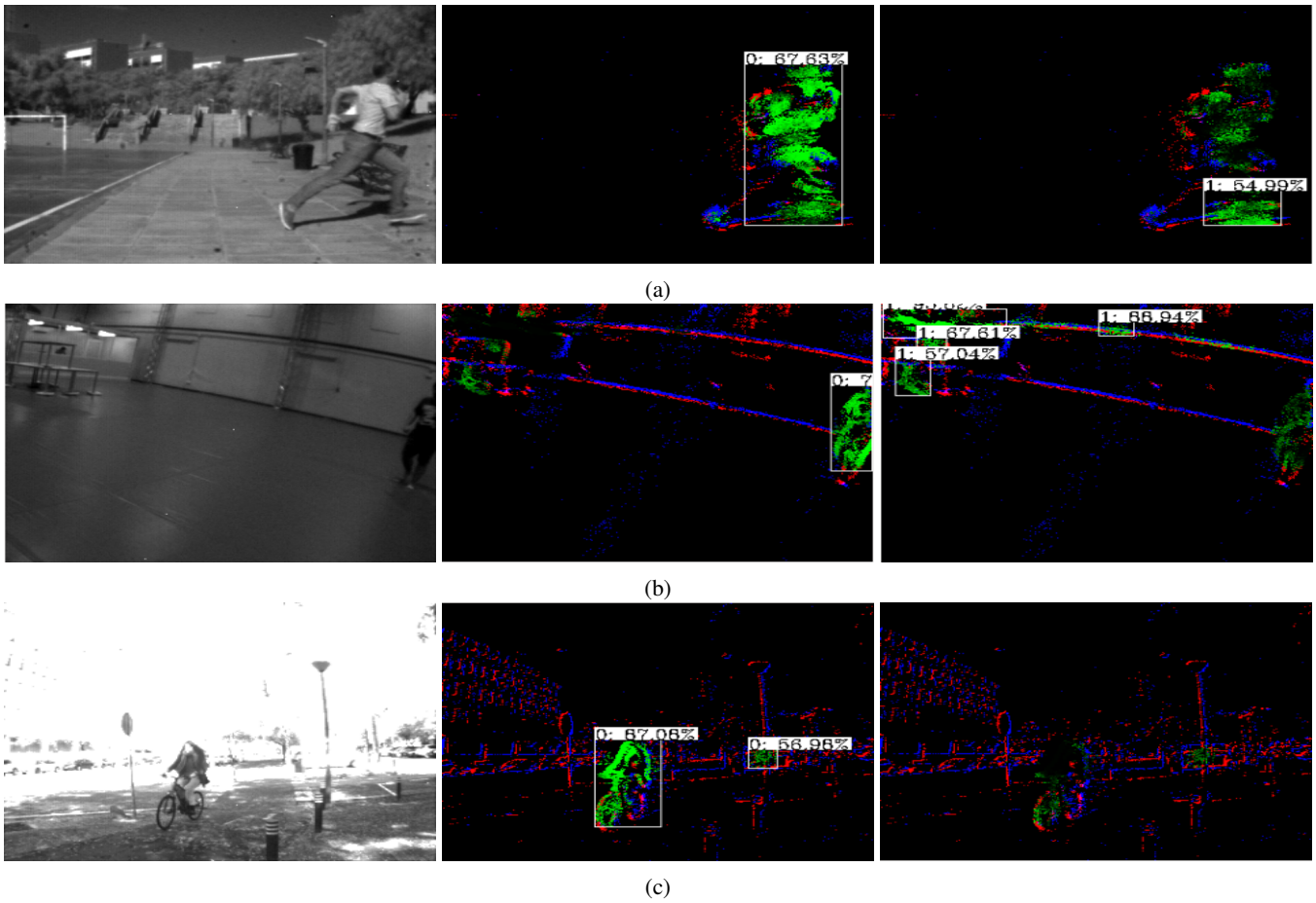


Fig. 8: Results of the trained CNN in some examples: a) moving person, b) person partially occluded, and c) person on a bike with overexposed lighting conditions. Each example shows: left) visual image, center) event image with events classified as `human` in green, right) event image with events classified as `non-human` in green. The events classified as the same class were clustered.

based surveillance scheme in Figure 2, i.e. the full proposed surveillance system, which is evaluated in Section V-B.

In our method α compensates for the differences in the representation of the different classes in the training datasets. Table II analyzes the influence of α on the CNN performance in terms of the rate of false positives, true negatives, false negatives, and true positives. Using w_i=1 provides FP=4.2%, TN=95.8%, FN=29.9%, and TP=70.1%. Although FP and TN are correct, FN and TP are unsuitable for the envisioned classification performance. It can observed that increasing the value of α reduces FP and TN, but increases FN and TP. A trade-off must be made to reduce FP but still maintaining a high value for TP. In our problem we adopted α=10 to have a low FP with a high TP.

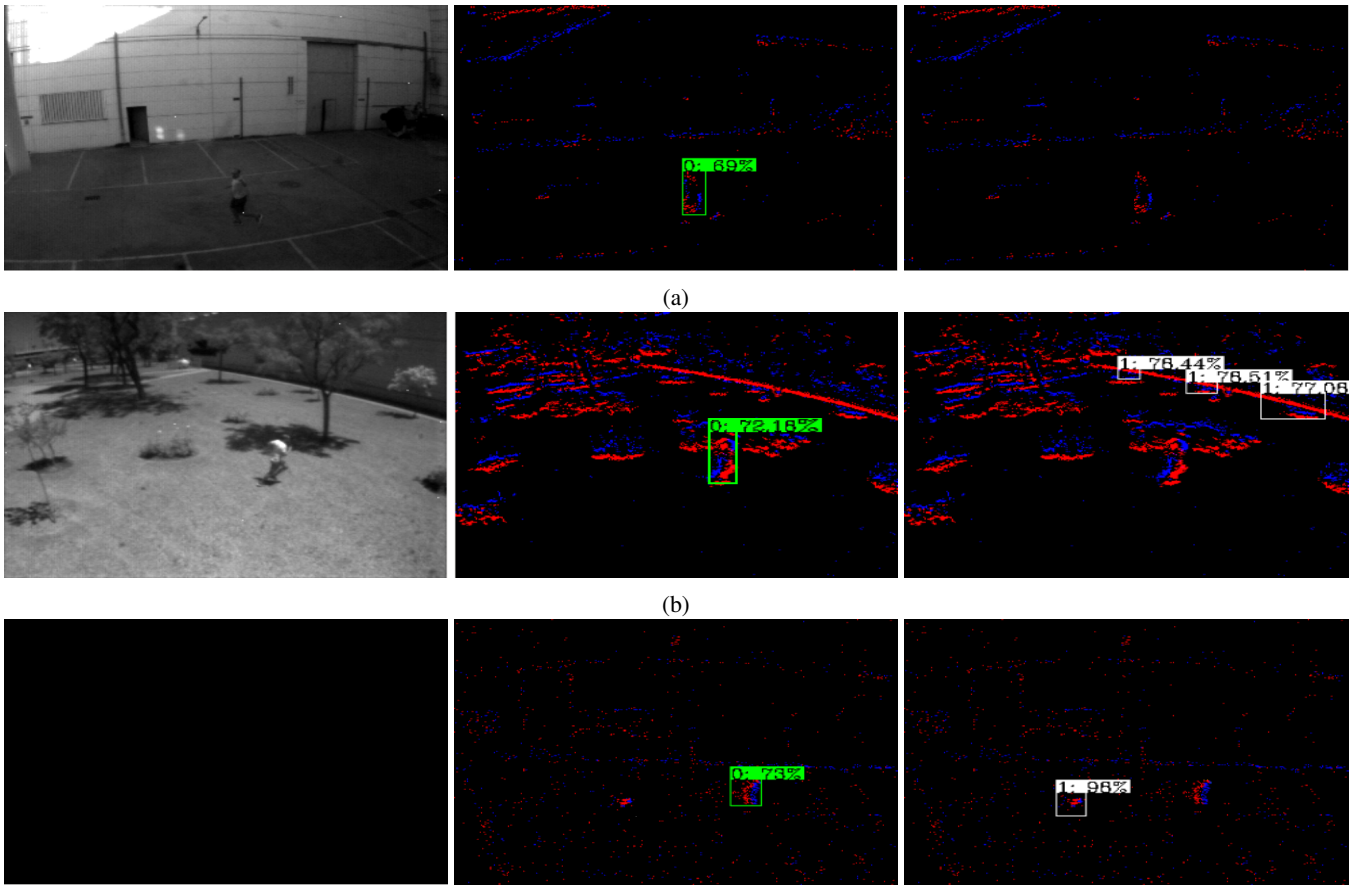
α value	FP(%)	TN(%)	FN(%)	TP(%)
1	16	84	10.3	89.7
5	14.3	85.7	11.6	88.4
10	13.9	86.1	12	88
20	10	90	15.1	84.9

TABLE II: Influence of α values in CNN performance. The adopted α=10 represents a trade-off between false positives (FP) and true positives (TP).

B. People detection for UAS surveillance

The full intruder detection scheme shown in Figure 2 is evaluated: *Intrusion Detector* analyzes the events from the event camera and detects clusters with consistent motion in the scene, while *Human Detector* processes solely the events assigned to a cluster detected by *Intrusion Detector*. This modified the percentages of events belonging to `human` and `non-human` in the classification problem, and the CNN had to be re-trained with the same training datasets but with a different value of α, i.e. α=5. Also, on board the UAS the objects appear on the *event images* with lower size. To take into account this effect the values of τ and η were reduced. The adopted values were τ=10^{-2} and η=18. The change in these parameters required the re-training of the CNN. Table III shows the performance of the re-trained CNN in the training and validation datasets. Although there is a little reduction in *Accuracy* and *Precision*, there is also an improvement in the *Recall*.

Next, the system performance was evaluated in complex



(a)

(b)

(c)

Fig. 9: Results of the full proposed detector in some examples: (a) daylight intrusion monitoring in a building area, (b) open field daylight intrusion monitoring, (c) nighttime intrusion monitoring. Each example shows: left) visual image, center) event image with green bounding box for `human`, right) event image with white bounding box for `non-human`.

Dataset	Accuracy(%)	Recall(%)	Precision(%)
training	86	90	68
validation	86	89	67

TABLE III: CNN performance with the training and validation datasets.

scenarios with different lighting conditions (including pitch dark conditions) and included cases with only one human intruder and also cases with one human intruder and several moving objects in the scene.

The validation results are analysed differently depending on the module of the proposed system. First, we obtain the metrics for the *Intrusion Detector* module, where TP, FP, TN and FN will be computed using as reference the number of clusters detected by *Intrusion Detector*, and the number of clusters that should be detected on the full video sequence. Then, we obtain the *Human Detector* module metrics using as reference only the clusters filtered by *Intrusion Detector*. Table IV shows the performance of each component, on the testing dataset. The *Accuracy* and *Recall* values of *Intrusion Detector* means that around 27% of clusters are mistakenly discarded. However, as it can be seen in the video[2], this have low impact on the full detection system. On the other hand, *Human Detector* performance is remarkable, obtaining more than 90% in all of the evaluated metrics. This highlights the benefits of mixing a proper pre-processing of the event stream with a deep learning classifier.

	Accuracy(%)	Recall(%)	Precision(%)
Intrusion Detector	72	73.1	90.6
Human Detector	93.4	95.9	92.7

TABLE IV: Performance of the testing dataset.

Next, we provide visual validation of our full system on different scenarios, including: one example with daylight intrusion monitoring in a building area (Figure 9-a); one example with daylight intrusion monitoring in an open field (Figure 9-b); one example of nighttime intrusion monitoring with several intruders (Figure 9-c). For visualization each example includes: the visual image from the DAVIS APS; event image where the clusters classified as `human` are surrounded by a green bounding box; and event image where

[2]https://youtu.be/Fc4X1TFmnVc

the clusters classified as `non-human` are surrounded by a white bounding box. The *event images* shown in red and blue colors the events of both polarities.

Figure 9 evidences the generalization capabilities of the proposed system. *Human Detector* is able to correct cases where *Intrusion Detector* creates clusters for non-moving objects, as in the example of Figure 9-b. In scenarios with more than one moving object, *Human Detector* outputs a correct classification, as in the example of Figure 9-c.

VI. Conclusions

This paper proposes an event-based processing scheme for human intrusion detection using UAS. It includes two main event-based processing systems. The first one detects intrusions as clusters of events with consistent motion. The second analyzes the events of the detected clusters using DL techniques (specifically, Convolutional Neural Networks) to compute the probability that the cluster corresponds to a person. Our method has two main novelties: 1) it is based purely on events, removing the need for other sensors onboard the UAS; and 2) the events are processed event-by-event fully exploiting the asynchronous capabilities of event cameras. The proposed scheme has been implemented and experimentally validated in challenging cluttered scenarios with different lighting conditions and different types of objects. Its performance validation shows values surpassing 90% in *Precision* and 70% in *Accuracy* and *Recall*, confirming its envisioned capabilities.

The use of event cameras for UAS-based surveillance applications is very promising. Enlarging the training dataset with a larger variety of scenarios and problems is recommended to enhance the system robustness. Besides, decreasing the computational cost to enable its implementation on low-cost hardware is object of current research. This can be achieved by re-designing the CNN structure and by further experimentally confirming the influence of the parameters involved in the *MITS* computation.

References

[1] J. P. Rodríguez-Gómez, A. Gómez Eguíluz, J. R. Martinez De-Dios, and A. Ollero, "Asynchronous event-based clustering and tracking for intrusion monitoring in uas," in *IEEE ICRA*, 2020.

[2] G. Gallego, T. Delbruck, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. Davison, J. Conradt, K. Daniilidis, *et al.*, "Event-based vision: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[3] S. Barua, Y. Miyatani, and A. Veeraraghavan, "Direct face detection and video reconstruction from event cameras," in *IEEE Winter Conf. on Applications of Computer Vision (WACV)*, 2016, pp. 1–9.

[4] Z. Jiang, X. Pengfei, K. Huang, W. Stechele, G. Chen, Z. Bing, and A. Knoll, "Mixed frame-/event-driven fast pedestrian detection," in *IEEE ICRA*, 2019.

[5] T. Stoffregen and L. Kleeman, "Simultaneous optical flow and segmentation (sofas) using dynamic vision sensor," *arXiv preprint arXiv:1805.12326*, 2018.

[6] A. Mitrokhin, C. Ye, C. Fermuller, Y. Aloimonos, and T. Delbruck, "Ev-imo: Motion segmentation dataset and learning pipeline for event cameras," *arXiv preprint arXiv:1903.07520*, 2019.

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

[8] A. Mitrokhin, C. Fermüller, C. Parameshwara, and Y. Aloimonos, "Event-based moving object detection and tracking," in *IEEE/RSJ IROS*, 2018, pp. 1–9.

[9] I. Alzugaray and M. Chli, "Asynchronous corner detection and tracking for event cameras in real time," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3177–3184, 2018.

[10] A. Gómez Eguíluz, J. P. Rodríguez-Gómez, J. R. Martinez De-Dios, and A. Ollero, "Asynchronous event-based line tracking for time-to-contact manuevers in uas," in *IEEE/RSJ IROS*, 2020.

[11] F. Barranco, C. Fermuller, and E. Ros, "Real-time clustering and multi-target tracking using event-based sensors," in *IEEE/RSJ IROS*, 2018, pp. 5764–5769.

[12] A. Z. Zhu, N. Atanasov, and K. Daniilidis, "Event-based visual inertial odometry," in *2017 IEEE CVPR*. IEEE, 2017, pp. 5816–5824.

[13] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258.

[14] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv*, 2018.

[15] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[16] A. Sironi, M. Brambilla, N. Bourdis, X. Lagorce, and R. Benosman, "Hats: Histograms of averaged time surfaces for robust event-based object classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1731–1740.

[17] D. Gehrig, A. Loquercio, K. G. Derpanis, and D. Scaramuzza, "End-to-end learning of representations for asynchronous event-based data," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 5633–5643.

[18] Y. Deng, Y. Li, and H. Chen, "Amae: Adaptive motion-agnostic encoder for event-based object classification," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4596–4603, 2020.

[19] X. Lagorce, G. Orchard, F. Galluppi, B. E. Shi, and R. B. Benosman, "Hots: a hierarchy of event-based time-surfaces for pattern recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 7, pp. 1346–1359, 2016.

[20] F. C. Ojeda, A. Bisulco, D. Kepple, V. Isler, and D. D. Lee, "On-device event filtering with binary neural networks for pedestrian detection using neuromorphic vision sensors," in *2020 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2020, pp. 3084–3088.

[21] D. P. Moeys, F. Corradi, E. Kerr, P. Vance, G. Das, D. Neil, D. Kerr, and T. Delbrück, "Steering a predator robot using a mixed frame/event-driven convolutional neural network," in *Intl. Conf. on Event-based Control, Communication and Signal Processing*, 2016, pp. 1–8.

[22] M. Iacono, S. Weber, A. Glover, and C. Bartolozzi, "Towards event-driven object detection with off-the-shelf deep learning," in *IEEE/RSJ IROS*. IEEE, 2018, pp. 1–9.

[23] N. J. Sanket, C. M. Parameshwara, C. D. Singh, A. V. Kuruttukulam, C. Fermüller, D. Scaramuzza, and Y. Aloimonos, "Evdodgenet: Deep dynamic obstacle dodging with event cameras," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 10 651–10 657.

[24] M. Cannici, M. Ciccone, A. Romanoni, and M. Matteucci, "Asynchronous convolutional networks for object detection in neuromorphic cameras," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 0–0.

[25] N. Messikommer, D. Gehrig, A. Loquercio, and D. Scaramuzza, "Event-based asynchronous sparse convolutional networks," in *European Conference on Computer Vision*. Springer, 2020, pp. 415–431.

[26] P. U. Diehl, D. Neil, J. Binas, M. Cook, S.-C. Liu, and M. Pfeiffer, "Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing," in *2015 International joint conference on neural networks (IJCNN)*. ieee, 2015, pp. 1–8.

[27] Y. Cao, Y. Chen, and D. Khosla, "Spiking deep convolutional neural networks for energy-efficient object recognition," *International Journal of Computer Vision*, vol. 113, no. 1, pp. 54–66, 2015.

[28] B. Ramesh, A. Ussa, L. Della Vedova, H. Yang, and G. Orchard, "Pcarect: An energy-efficient object detection approach for event cameras," in *Asian Conference on Computer Vision*. Springer, 2018, pp. 434–449.

[29] A. Ollero and L. Merino, "Control and perception techniques for aerial robotics," *Annual reviews in Control*, vol. 28, no. 2, pp. 167–178, 2004.

[30] R. Hudson, *Infrared System Engineering*, ser. Wiley series in pure and applied optics. Wiley-Interscience, 1969.

[31] A. Torres-González, J. R. Martinez-de Dios, and A. Ollero, "Range-only slam for robot-sensor network cooperation," *Autonomous Robots*, vol. 42, 07 2017.

[32] J. R. Martinez-de Dios et al., "GRVC-CATEC: aerial robot co-worker in plant servicing (ARCOW)," in *Bringing Innovative Robotic Technologies from Research Labs to Industrial End-users*, ser. Springer Tracts in Advanced Robotics, F. Caccavale, C. Ott, B. Winkler, and Z. Taylor, Eds. Springer, 2020, vol. 136, pp. 211–242.

[33] 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 Workshop on Unconventional Sensors in Robotics*. IEEE, 2020.

[34] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising," *IEEE transactions on image processing*, vol. 26, no. 7, pp. 3142–3155, 2017.

[35] Q. Chu, W. Ouyang, H. Li, X. Wang, B. Liu, and N. Yu, "Online multi-object tracking using cnn-based single object tracker with spatial-temporal attention mechanism," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 4836–4845.

[36] H. Rebecq, R. Ranftl, V. Koltun, and D. Scaramuzza, "Events-to-video: Bringing modern computer vision to event cameras," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3857–3866.

[37] T. Stoffregen, C. Scheerlinck, D. Scaramuzza, T. Drummond, N. Barnes, L. Kleeman, and R. Mahony, "How to train your event camera neural network," *arXiv preprint arXiv:2003.09078*, 2020.

[38] I. Alonso and A. C. Murillo, "Ev-segnet: Semantic segmentation for event-based cameras," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 0–0.

[39] D. Zou, F. Shi, W. Liu, J. Li, Q. Wang, P.-K. Park, C.-W. Shi, Y. J. Roh, and H. E. Ryu, "Robust dense depth map estimation from sparse dvs stereos," in *British Mach. Vis. Conf.(BMVC)*, vol. 1, 2017.

[40] B. Ramesh, H. Yang, G. Orchard, N. A. Le Thi, S. Zhang, and C. Xiang, "Dart: distribution aware retinal transform for event-based cameras," *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, no. 11, pp. 2767–2780, 2019.

[41] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*. MIT press Cambridge, 2016, vol. 1, no. 2.

[42] J. Davis and M. Goadrich, "The relationship between precision-recall and roc curves," in *Proceedings of the 23rd Int. Conf. on Machine learning*. ACM, 2006, pp. 233–240.