Received June 23, 2020, accepted June 28, 2020, date of publication July 1, 2020, date of current version July 20, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3006191

Adaptive Inattentional Framework for Video Object Detection With Reward-Conditional Training

ALEJANDRO RODRIGUEZ-RAMOS, (Graduate Student Member, IEEE), JAVIER RODRIGUEZ-VAZQUEZ, CARLOS SAMPEDRO, (Member, IEEE), AND PASCUAL CAMPOY, (Senior Member, IEEE)

Centre for Automation and Robotics, Computer Vision and Aerial Robotics Group, Universidad Politécnica de Madrid (UPM-CSIC), 28006 Madrid, Spain

Corresponding author: Alejandro Rodriguez-Ramos (alejandro.rramos@upm.es)

This work was supported in part by the Spanish Ministry of Economy and Competitivity through the project (Complex Coordinated Inspection and Security Missions by UAVs in cooperation with UGV) under Grant RTI2018-100847-B-C21, in part by the MIT International Science and Technology Initiatives (MISTI)-Spain through the project (Drone Autonomy), and in part by the Mohamed Bin Zayed International Robotics Challenge (MBZIRC) in the year 2020 (MBZIRC 2020 competition).

ABSTRACT Recent object detection studies have been focused on video sequences, mostly due to the increasing demand of industrial applications. Although single-image architectures achieve remarkable results in terms of accuracy, they do not take advantage of particular properties of the video sequences and usually require high parallel computational resources, such as desktop GPUs. In this work, an inattentional framework is proposed, where the object context in video frames is dynamically reused in order to reduce the computation overhead. The context features corresponding to keyframes are fused into a synthetic feature map, which is further refined using temporal aggregation with ConvLSTMs. Furthermore, an inattentional policy has been learned to adaptively balance the accuracy and the amount of context reused. The inattentional policy has been learned under the reinforcement learning paradigm, and using our novel reward-conditional training scheme, which allows for policy training over a whole distribution of reward functions and enables the selection of a unique reward function at inference time. Our framework shows outstanding results on platforms with reduced parallelization capabilities, such as CPUs, achieving an average latency reduction up to 2.09×, and obtaining FPS rates similar to their equivalent GPU platform, at the cost of a 1.11× mAP reduction.

INDEX TERMS Inattention, YOTO, reward-conditional training, deep learning, video object detection, reinforcement learning, CNN, LSTM, loss-conditional training.

I. INTRODUCTION

Recent advances in image object detection have mainly focused on the development of Convolutional Neural Network (CNN) architectures [1]–[4] to progressively increase accuracy or decrease processing times. In this regard, accuracy has been the primary concern in the majority of studies [3], [5]–[7], mostly due to the initial lack of techniques which precisely capture the variability found in the object detection task (number of classes, illumination and environmental ambiences, corner cases, etc.). Nevertheless, detection speed and power consumption are recently becoming key differentiator metrics [1], [8]–[13] as deep learning is being increasingly deployed in practical applications, where computing and power resources can be limited. In the context of systems with reduced computational capabilities, such as embedded platforms, substantial effort has been carried out to generate efficient architectures, such as the ones based on SqueezeNet [14], Mobilenet [15]–[18], or ShuffleNet [19], [20]. Also, other studies target efficient representations, such as Binarized Neural Networks (BNNs) [21] or Quantized Neural Networks (QNNs) [22], and low-power hardware implementations [23]–[25]. Despite significant advances in the field, the final objective of processing high-resolution images in real-time, on embedded systems and without notable accuracy loss, is still an open problem.

The associate editor coordinating the review of this manuscript and approving it for publication was Ting Wang.

However, human vision system provides intuition on how such solution can be achievable. On this basis, human vision has been found as foveal and active [26], [27]. Environment observation at high-resolution is achieved by the fovea, and only corresponds to 5 degrees of the total visual field [28]. Our complementary peripheral vision, which processes observations at a lower resolution, has a field of view of 110 degrees [29]. To track objects, our eyes perform quick saccadic movements, followed by smooth pursuit movements [30], [31]. In this context, when the focus is on an object, its peripheral surroundings can be simply ignored, a phenomenon known as inattentional blindness [32]. Hence, detecting and tracking objects is an active process, and the amount of information processed depends on the complexity and the dynamism of the scene, as well as on the attention of the observer. Besides, a fundamental advantage of human vision is that it relies on a stream of images, rather than on single images. Humans can rely on contextual cues and memory to supplement their understanding of the image. On this subject, some of the stated properties of human vision have led to novel neuromorphic sensor designs, such as Dynamic Vision Sensors (DVS), which react to events and decrease the amount of information processed. Although DVS-related research studies are still in early stages, they have provided promising results in the context of asynchronous object detection and tracking at high-speeds and low-power consumption [33], [34]. Conversely, this work, where a standard RGB camera sensor has been utilized, follows the idea of decreasing the amount of data to be processed, and examines the question of whether neural networks are similarly able to dynamically neglect image information when assisted by a memory and previous image context, providing a lower computation overhead.

An important property of video is that adjacent video frames are highly correlated, which opens up the possibility of decreasing the computation latency. During the process of object detection in a sequence, surrounding context of previous frame detection is prone to hold redundant content, which can be exploited in the current detection stage. Therefore, a simple idea is to keep a memory of previous extracted features and recompute only the ones corresponding to the region where the object was found in previous frame. To follow this idea, features from previous and current frame have to be fused in a synthetic feature map, which is used to compute the final detection. In this trend, based on the intuition that human peripheral vision may not add value while pursuing an object, our approach dynamically reuses feature context from previously detected frames to increase efficiency.

From these observations, we propose a simple but effective pipeline for efficient video object detection, illustrated in Fig. 1. Concretely, we introduce a novel inattentional framework, where peripheral detection context is dynamically reused to speed up inference latency while maintaining accuracy. The context features corresponding to keyframes are fused into a synthetic feature map, which is further refined

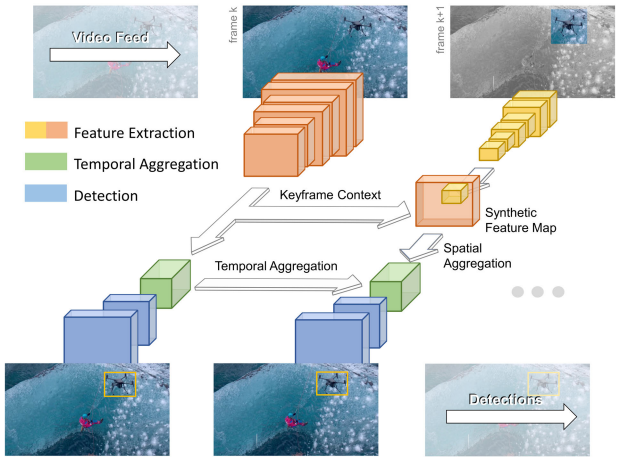

FIGURE 1. A schematic diagram of the proposed inattentional framework. The keyframe context is dynamically reused to reduce computation overhead, and temporal information is propagated by the temporal aggregator. Grey region in an image frame denotes pixels which are not being processed for the given frame.

based on its temporal structure. The temporal aggregation is carried out by a Convolutional Long-Short Term Memory (ConvLSTM) layer, and detections are generated by fusing context from previous frames with the new information of the current frame. Additionally, we show that the aggregation of our synthetic feature maps by the ConvLSTM contains within itself the information necessary to decide when the peripheral context has to be ignored. We learn an *inattentional policy* of when to use the full or synthetic feature map by formulating the task as a reinforcement learning problem.

In addition, the amount of attention a human observer gives to an object in a video sequence can be dynamic, modulating the inattentional blindness based on the one individual intentions. On the side of deep learning, neural networks are static structures which response commonly remains invariant during inference. This is due to the fact that during common network training stages, the parameters of a given loss function are not normally altered, narrowing the behavior of the network to a unique state of the potential distribution of loss functions. In this regard, recent advances reveal that a neural network can be trained not only over a unique loss function but over a distribution of them, as in You Only Train Once (YOTO) [35], [36]. Following this direction, the present work explores the possibility of extending this idea to the reinforcement learning paradigm, where the standard policy training involves defining a unique performance metric [37]–[39]. Under the context of the proposed application, the behavior of the inattentional policy network has been extended to dynamically provide different attention ratios, based on an inattentional parameter, which modulates the target performance metric on inference time.

While prior works, mostly flow [40]–[47] and recurrent [43], [48]–[52] methods, also provide approaches for fast video object detection with fast and slow detection stages, these approaches are based on processing the full image

for every frame, incurring in a redundancy of computation and not exploiting the performance increase when the object appears reduced in the image. Conversely, our method takes advantage of the redundancy of context in video frames, increasing the performance when objects appear small in the image. Furthermore, approaches based on the reinforcement learning framework [49], provide policies trained over a unique performance metric *i.e.* reward function. Our policy network has been trained over a distribution of reward functions, providing not only a unique behavior but a distribution of them on inference time. Our method has been validated on the Imagenet VID 2015 [53] dataset and on our custom dataset, which has been made publicly available.

In summary, the contributions of the present work are as follows:

- We present a human-inspired inattentional framework for video object detection, where context features are dynamically reused in a synthetic feature map in order to reduce redundant computation, and their outputs are fused using a recurrent memory module.
- We introduce an adaptive inattentional policy where the decision over the context features computation is learned with deep reinforcement learning, which leads to a higher speed-accuracy trade-off.
- We demonstrate a successful application of YOTO [35] to the reinforcement learning paradigm, where a policy has been trained over a distribution of reward functions. To the authors knowledge, this is the first time this technique is applied to the reinforcement learning framework.
- The custom dataset used for validation of the framework has been made publicly available.

The remainder of the paper is organized as follows: Section III describes our approach and provides a detailed explanation about the inattentional fundamentals and the reward-conditional training, explaining the problem formulation and the system architecture. Section IV-B outlines the carried-out experiments and their corresponding results and Section V remarks on and discusses the most relevant experimentation outcomes. Finally, Section VI concludes the paper and indicates future lines of research.

II. RELATED WORK

This work proposes a video object detection framework and, to this aim, is aided by techniques of diverse nature, such as temporal aggregation or keyframe selection methods. As a consequence, there are several related works that are adjacent to the case under study.

A. VIDEO POST-PROCESSING METHODS

Early research for extending single-image object detection to the video domain was commonly focused on the generation of frame detection tracks, where current detection is linked to the previous ones in the track. Through these tracks, previous information is aggregated to improve the current frame object detection accuracy.

The Seq-nms algorithm proposed in [54] applies dynamic programming to find tracks and improve the confidence of weaker detections. Tubelets with Convolutional Neural Netowrks (TCNN) [55], [56] proposes a pipeline for detection propagation across frames via optical flow, and a tracking algorithm to refine scoring by finding tubelets. These initial strategies yielded accountable improvements in performance, but did not fundamentally change the basic per-frame detection techniques. In our work, we take advantage of the key idea of aiding current detection with previous detection context to decrease the computation time, while maintaining the accuracy.

B. FEATURE AGGREGATION OVER TIME

One key aspect which differentiates video from single-image detection, is the existence of encoded information that remains low-variant across several video frames. Consequently, there exist inter-frame features which can be aggregated and exploited to improve performance. In this regard, multiple techniques for stated feature aggregation have been explored in the literature.

In [40], intermmediate feature maps in a CNN were able to be temporally propagated across frames by means of optical flow. The Deep Feature Flow (DFF) framework [40] allows for feature propagation across multiple frames in order to compute detections on sparse keyframes, which alleviates the computational cost. Flow-Guided Feature Aggregation (FGFA) [41] further explored this idea by warping and averaging features from adjacent frames, which lead to acurracy improvements. Impression networks [42] provided a computation reduction by combining sparse ''impression features'', which stores low-variant and long-term information, with optical flow and warping aggregation. In [43] an efficient feature flow aggregation for different keyframe selection schemes is introduced. The Flow-Track framework [44] combines historical feature flows, warping, consine similarity and temporal attention for efficient feature aggregation achieving an increased performance. Some techniques, primarily meant for semantic object segmentation [45]–[47], also utilized warped feature flow maps to aid the final stages of the segmentation pipeline.

Recurrent architectures, such as Gated Recurrent Units (GRUs) or LSTM cells, have been integrated to enable the process of feature aggregation. In [43], a flow-guided GRU is proposed to effectively aggregate features on keyframes. Spatial-Temporal Memory Modules (STMMs) [48] were proposed to make better use of single-image detector weights pre-trained on large scale image datasets. ConvLSTMs variants have been more widely utilized for feature aggregation across frames [49]–[52]. Although ConvLSTM can fundamentally serve as a memory which is able to store mid-term information across frames [50], [51], it can be further extended to aggregate features from different extractors [49] or from an encoding of multiple frames [52].

Our ConvLSTM memory module is similar to [51], but serves an additional purpose of fusing features from real and synthetic feature maps, posing an additional challenge.

Furthermore, external memories can benefit the long-term information storage, which can be useful for feature aggregation in the video domain [57], [58]. Besides, some techniques integrate detection trackers to exploit temporal information between keyframe processing [59]–[61]. Other strategies rely on Locally-Weighted Deformable Neighbors (LWDNs) [62], Spatio-Temporal Sampling Networks (STSNs) [63], Motion History Image (MHI) [64], spatially variant convolutions [65] or cross-correlations [66] for feature propagation.

C. ADAPTIVE KEYFRAME SELECTION

Multiple approaches interleave processing pipelines with different computational budgets during the execution of the video object detection, in order to provide both fast and accurate solutions. This procedure commonly involves keyframe selection, where stated keyframe is processed by a more expensive but accurate pipeline. Despite keyframe selection can be carried out naively (*i.e.* fix rate), providing proper solutions [40], [43], [66], a more adaptive method for the case under study is potentially able to improve performance. For instance, to adapt to video input sources of variable complexity.

In [61], the optimal combination of detection and tracking stages is found in an offline trend for a specific video. In [43], a feature consistency indicator was used in a heuristic rule to select keyframes. A keyframe selection policy has been also incorporated to standard supervised learning schemes, such as the usage of low-level features for predicting rapid changes in the input [65], increase of complexity [45] or features quality [42].

Other techniques take advantage of the reinforcement learning framework for adaptive keyframe selection [47], [49], [60]. A policy-gradient reinforcement-learning approach [47] makes budget-aware processing by approximating the gradient of a non-decomposable and non-differentiable objective. In [67], a learned policy is able to select when to update or to initiate a tracker. In [60], the reinforcement learning policy is guided to achieve a proper balance between detection and tracking. In [49] a light policy is used for balancing the execution pipeline between an expensive and a cheap feature extractor, achieving better results as compared to a random baseline. Our keyframe selection strategy has been inspired by the adaptive keyframe selection method found in [49]. In this direction, an adaptive selection policy has been learned within the reinforcement learning framework in order to select between real and synthetic feature maps (expensive and cheap pipeline, respectively) for detection. Furthermore, our complete policy has been trained over a distribution of reward functions, instead of a unique and constrained reward function, achieving proper performance throughout the whole domain.

D. SINGLE-IMAGE RoI SELECTION METHODS

A feed of video images opens the possibility of a potential increment of efficiency due to information redundancy across frames. Nevertheless, within the single-image context, there also exist regions which do not contain useful information to be exploited during the object detection task. In this regard, a notable amount of techniques have been explored in the literature. Stated strategies primarily aim at reducing the amount of information processed in an single image by selecting Regions of Interest (RoIs).

The generation of saliency maps can help during the RoI selection procedure. In the context of classical computer vision, log-spectrum [68], as well as a combination of image filters [69] (color, intensity, and orientation filters, among others) have been utilized for saliency computation. Additionally, RoI selection can be automated with standard supervised learning techniques. AutoFocus [70] forwards the full image through a cheap network to generate ''chips'' which are further processed in higher resolutions. In [71], a cropped image and its context is used in order to improve small object detection. AZ-Net [72] method zooms recursively into RoIs, which are likely to contain objects. In [73], RoIs are generated throughout an iterative process by the network, and a voting process is carried out to select the final regression. In [74], semantic and context information is used to find RoIs.

On the other hand, reinforcement learning has been notably applied to the problem of RoI selection. Dynamic zoom-in network [9] inputs a full low-resolution image and uses a reinforcement learning policy to select RoI to be further fine processed. In [75], an LSTM-based model is trained with reinforcement learning to give attention to certain RoI of the image in order to find small objects. In [76], [77], a reinforcement learning based sliding window approach finds an object in a few steps. Reinforcement learning based Region Proposal Network (RPN) [10] has also been explored to increase its efficiency of computation. In [78], [79], an agent selects the regions of the image which need to be processed at higher resolution or by a finer detector to reduce computation. In [80] RoIs are effectively selected based on the interdependence of objects with tree-structured reinforcement learning. In this work, RoIs are strongly related to the previous keyframe detection at a certain instant of time. Also, RoI size decision is based on the state of our ConvLSTM, being potentially assisted by the context of previous detected object in the sequence.

III. THE INATTENTIONAL FRAMEWORK

In this section, the proposed framework is described. First, the inattentional fundamentals are explained. Then, the adaptive inattentional policy as well as the reward-conditional training procedure are described in detail. The nomenclature and abbreviations used in the description of the inattentional framework are summarized in Table 1.

A. INATTENTIONAL FUNDAMENTALS

Following the intuition provided by the inattentional blindness phenomenon [32], our solution is approached as a

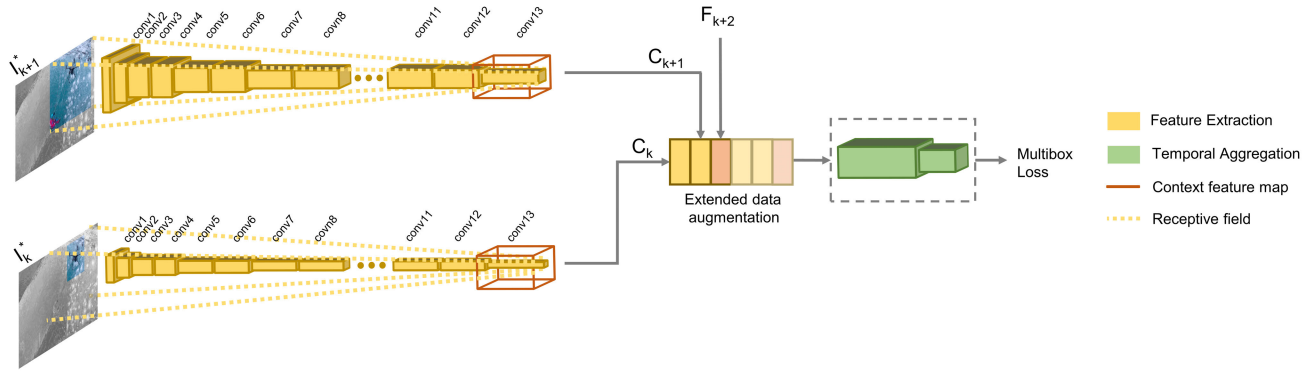

FIGURE 2. Structure of the extended data augmentation technique. The architecture shown corresponds to MobileNetV1 feature extractor. The receptive field in a deep layer of a standard CNN is usually a notable region of the input image. In order to further reduce the computation overhead, image crops I^*_k have not been selected to match the required receptive field. To tackle this issue, the resulting feature maps C_k have been included in the augmented training data for generality of the temporal aggregator.

TABLE 1. Table of abbreviations and nomenclature in the context of the proposed inattentional framework.

Symbol	Definition
f	Feature extractor
c	Context aggregator
a	Temporal aggregator
d	Detector
$\mathcal{V}$	Video sequence
I_k	k-th image frame
I^*_k	k-th inattentional image frame (image crop)
F^c_k	Feature map context (output from **f**)
F^*_k	Feature map of inattentional frame (output from **f**)
C_k	Synthetic feature map (output from **c**)
A_k	Temporally aggregated feature map (output from **a**)
D_k	Detections (output from **d**)
λ_0	Reward-conditional parameter

dynamic context reuse across an image sequence $\mathcal{V} = \{I_0, I_1, I_2, \ldots, I_n\}$. Indeed, the phenomenon of momentarily ignoring context, while detecting an object, can be effectively reformulated as a context reuse, since reusing previous context is equivalent to avoiding current context computation. Besides, our framework is restricted to the online setting where only $\{I_0, I_1, \ldots, I_k\}$ are available during the computation of the k-th detection.

In order to materialize this concept into the deep learning paradigm, four main components have been defined in the present framework: a *feature extractor* **f**, a *context aggregator* **c**, a *temporal aggregator* **a** and a *detector* **d**. The context is defined as a subset of the feature maps generated by the feature extractor **f** on a keyframe k. In this direction, our approach interleaves full keyframe computation and partial frame (inattentional frame) computation with context aggregation (see Fig. 2 and Fig. 3) to provide an intermmediate representation. The frame-level features are then temporally aggregated and refined using a recurrent architecture. Finally, an SSD-style [17] detection pipeline is applied on the refined feature map to obtain bounding box results.

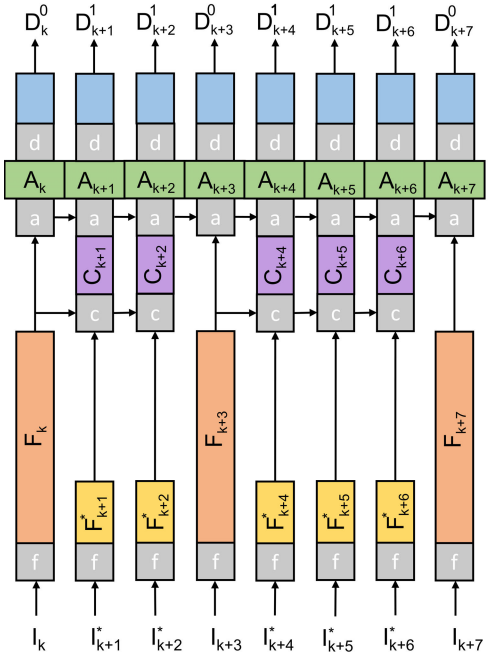

FIGURE 3. A block diagram of our adaptive keyframe selection technique with all the components of the inattentional framework included, where *f* stands for feature extractor, *c* is the context aggregator, *a* depicts the temporal aggregator and *d* represents the detector. For clarity, coloured blocks correspond to tensors and grey blocks correspond to components of the system. The length of the tensor denotes time consumption but it is not to scale. The context of a keyframe F^c_k is not being modified throughout the pipeline of context aggregators.

Each step of the processing pipeline can be defined as a function mapping. The feature extractor **f**: $\mathbb{R}^I \rightarrow \mathbb{R}^F$, maps the image space into an encoded feature space $\mathbb{R}^F$. The context aggregator **c**: $\mathbb{R}^F \times \mathbb{R}^F \rightarrow \mathbb{R}^C$ optionally composes a synthetic feature map C_k, based on previous keyframe context F^c_k and current reduced feature map F^*_k. The temporal aggregator **a**: $\mathbb{R}^F \times \mathbb{R}^S \rightarrow \mathbb{R}^A \times \mathbb{R}^S$ or $\mathbb{R}^C \times \mathbb{R}^S \rightarrow \mathbb{R}^A \times \mathbb{R}^S$, refines the generated feature map (either full $\mathbb{R}^F$ or synthetic $\mathbb{R}^C$) based on previous temporal information cues, compute its state $\mathbb{R}^S$

and outputs an updated feature space $\mathbb{R}^A$. The SSD detector $\mathbf{d}$: $\mathbb{R}^A \rightarrow \mathbb{R}^D$, maps the aggregated feature space into final detection predictions. In order to obtain detections D_k, one may compute A_k by $\mathbf{a}(\mathbf{f}(I_k), s_{k-1})$ or $\mathbf{a}(\mathbf{c}(\mathbf{f}(I_k^*), F_k^c), s_{k-1})$, where s_{k-1} is the previous frame state and, I_k and I_k^* are the full image or cropped image for the k-th frame, respectively. The final detections D_k are obtained as $D_k = \mathbf{d}(A_k)$.

The context aggregator $\mathbf{c}$ is a key component in the inattentional framework, as it is in charge of reusing context from previous keyframes, in order to provide a more efficient performance while reducing computation overhead. Specifically, reusing context requires fusing the current feature map F_k^* with the context feature map F_k^c. Since the context is related to the object in the image space, one simple approach for fusing feature maps is to calculate the precise position of a feature map corresponding to an image crop I_k^* inside a context feature map F_k^c corresponding to a full keyframe I_{k-n}. Furthermore, in order to precisely select the context crop in the image space (image crop I_k^*), previous detection D_{k-1} and the particular receptive field of F_k have to be taken into consideration. However, the receptive field of a given layer unit is crucial to generate a feature map F_k^* which can be directly placed (without distortion) in the context feature map F_k^c. In this respect, it is important to highlight that the receptive field size of a deep layer in a neural network normally covers a notable percentage of the input image (the receptive field size of the final layer increases with the depth and the striding of the network). At this point, a trade-off, between the image crop I_k^* size and the computation overhead stands out, since in order to account the full size of the receptive field, the image crop can be increased for a certain object, incurring in more computation overhead, and vice versa.

As the present framework targets computation efficiency, the amount of aggregated context has to be maximized in order to reduce the computation overhead. This fact leads to the generation of image crops I_k^* with a small size which may not be matching the required receptive field for the feature map F_k (see Fig. 2). In this scenario, the aggregated feature map C_k can exhibit artifacts around the area where F_k^* was placed, resulting in a lower accuracy of the complete approach. To tackle this issue, the temporal aggregator $\mathbf{a}$ has been also trained to adapt $\mathbb{R}^C \rightarrow \mathbb{R}^F$ domains via extended feature map augmentation during training. Thus, the temporal aggregator has been trained with augmented data which aids the detection with synthetic feature maps (refer to Training Methodology in Section IV-A). In this context, the temporal aggregator not only aggregates temporal video information, but also refines feature maps from different nature (full and synthetic).

B. ADAPTIVE INATTENTIONAL POLICY

Although context aggregation at random intervals is able to provide competitive results in terms of latency reduction while maintaining accuracy (refer to Section IV-B), a natural question is whether an adaptive policy can improve these results by using the state of the system as well as the complexity of the input as information sources. In this regard, a variable amount of inattention can be provided to the context, based on the uncertainty of past detections and the object dynamism in a given sequence. Accordingly, a novel adaptive inattentional policy has been proposed, using formulation of the reinforcement learning paradigm.

In reinforcement learning, an agent interacts with an environment, seeking to find the maximum accumulated reward over time. To formulate the reinforcement learning problem, it is necessary to define an action space $\mathbb{A}$, a state space $\mathbb{S}$, and a reward function r. In the proposed inattentional framework, the inattentional policy has the ability of executing two actions: it can decide whether to execute the full expensive pipeline or, on the contrary, to select the faster context aggregation pipeline (with the potential loss on accuracy). In this regard, the discrete action space $\mathbb{A}$ is defined as $\mathbb{A} = a$ with $a \in \{0, 1\}$.

Additionally, our proposed policy is meant to examine the temporal aggregator state in order to find insights about the uncertainty of the detection, as well as about abrupt context changes in order to perform the best possible action, leading to an optimum policy $\pi^*(s_t|\theta)$. Following this idea, the state space $\mathbb{S}$ is defined as

$$\mathbb{S} = (c_k, h_k, c_k - c_{k-1}, h_k - h_{k-1}, \rho_k, b_{k-1}, \psi) \qquad (1)$$

where c_k and c_{k-1} are the current and previous ConvLSTM cell states, respectively; h_k and h_{k-1} are the current and previous ConvLSTM hidden states, respectively; ρ_k is an action history vector and has been empirically found to work properly with size 20, b_{k-1} is the normalized previous I_k^* RoI (it is the normalized full image RoI if previous frame was a keyframe), and ψ is the inattentional factor. The inclusion of $(c_k - c_{k-1})$ and $(h_k - h_{k-1})$ helps detecting changes in the temporal aggregator state. Also, ρ_k and b_{k-1} make the agent aware of its previous actions and context size, in order to avoid excessively running the context aggregation pipeline. The size of the state with the current architecture is 102,425 continuous variables.

The reward function r is a crucial component in the reinforcement learning framework. Indeed, it can either speed up training or, conversely, a naive design can completely prevent the agent from learning. Our reward must reflect the intention of finding balance between running the context aggregation pipeline as much as possible while maintaining accuracy. In this work, the reward function has been adapted from [49] as

$$r = \begin{cases} \min\limits_i (L(D^i)) - L(D^0) & a = 0 \\ \psi + \min\limits_i (L(D^i)) - L(D^1) & a = 1 \end{cases} \qquad (2)$$

where D^0 and D^1 are the detection results through the expensive pipeline (full image) and cheap pipeline (context aggregation), respectively; $L(\cdot)$ is the Multibox loss [81] computation and ψ corresponds to the inattentional factor. The inattentional factor ψ is an important component of the reward function. It is a scalar value which can potentially

encourage the agent taking the cheap pipeline, even when its cost remains higher. The definition of the ψ value is decisive for the final behavior of the agent.

C. REWARD-CONDITIONAL TRAINING

Conventional methods for training machine learning algorithms define one or more differentiable cost functions to perform the gradient propagation to the input(s). Typically, cost functions include static parameters which are able to modulate training and/or inference performance, leading to a static behavior of the network at inference time. Stated cost function parameters can be tuned based on experience, trial-and-error, grid search or more advanced methods. However, recent techniques have revealed a convolutional network can be trained over the whole distribution of parameterized cost functions, as shown in (3), enabling the possibility of conditioning the network on a subset of parameters at inference time (YOTO [35]).

$$\theta^* = \underset{\theta}{arg\,min} \sum_{i=1}^{n} L(\mathbf{y}_i, F(\mathbf{x_i}, \theta, \lambda_\mathbf{i}), \lambda_i), \quad \lambda_i \sim P_\lambda \tag{3}$$

where L represent one of the n cost functions, λ_i are the cost function(s) parameters and θ the model parameters. Also, in [35], Feature-wise Linear Modulation (FiLM) [36] is used to condition the network on the loss parameters. This technique has been applied in [35] to fully convolutional models in a supervised and semi-supervised trend, *e.g.* β-variational autoencoder, image compression and style transfer.

In this work, we extend the loss-conditional training framework [35] by proposing a novel reward-conditional training, under the reinforcement learning formulation. As in the supervised or semi-supervised case, a common agent in reinforcement learning is trained over a parameterized performance function, in this case a reward function, which has to be properly adjusted to encourage the desired behavior. Once the reward function and its parameters are defined, the agent behavior converges to a subset of possible behaviors within the distribution of the reward function parameters. Instead of defining a unique parameterized reward function, we propose to condition the network policy $\pi^*(s_t|\theta, \lambda_i)$ on the whole distribution of parameterized reward functions, as in (4) and (5).

$$\pi^*(s_t|\theta, \lambda_i) = \underset{a_t \in \mathbb{A}}{arg\,max}\ Q(s_t, a_t|\theta, \lambda_i), \quad \lambda_i \sim P_\lambda \tag{4}$$

$$\begin{aligned} Q(s_t, a_t|\theta, \lambda_i) &= \mathbb{E}_\pi[R_t(\lambda_i)] \\ &= \mathbb{E}_\pi[\sum_{k=0}^{\infty} \gamma_r^k r(\lambda_i)_{t+k+1}] \end{aligned} \tag{5}$$

where θ corresponds to the model function parameters, λ_i are the reward-conditional parameters, γ_r is the discount factor and $r(\lambda_i)$ denotes the distribution of conditioned reward functions.

For the case under study, our reward function is composed of one important parameter $\psi = \lambda_0$, which balances the amount of accuracy an agent is able to sacrifice when running both expensive and cheap pipelines. λ_0 has been selected as the reward-conditional parameter. Analogously to the reward expression shown in (2), the formulation of the final conditioned reward function is illustrated in (6) for clarity.

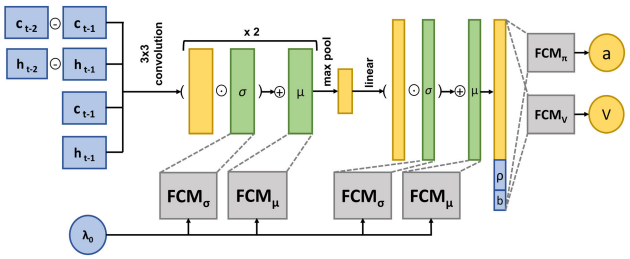

FIGURE 4. The architecture of the inattentional policy network. The state tensors have been depicted in dark blue, as well as the λ_0 reward-conditional parameter. Part of the state (ρ and b) is being appended in the last fully connected layer of the network. The reward-conditional parameter has been integrated by applying FiLM technique to the tensors represented in dark green.

$$r(\lambda_0) = \begin{cases} \min\limits_i (L(D^i)) - L(D^0) & a = 0 \\ \lambda_0 + \min\limits_i (L(D^i)) - L(D^1) & a = 1 \end{cases} \tag{6}$$

where $\lambda_0 \sim P_\lambda$ and P_λ denotes a log uniform distribution of probability with $\lambda_0 \in [0, 2]$.

In our experiments, the policy $\pi(s_t|\theta, \lambda_0)$ is built on a light convolutional backbone and two Fully Connected Models (FCMs), which output both the action and the value (see Fig. 4). To condition the network on the reward parameters, we use FiLM. We have selected the convolutional layers and one fully-connected layer to be conditioned by λ_0. Assume a given feature map $\mathbf{f}$ of dimensions $W \times H \times C$, with W and H corresponding to the spatial dimensions and C to the channels. The scalar value λ_0 is fed to two FCMs, M_σ and M_μ, to generate two vectors, σ and μ of dimensionality C each. We then multiply the feature map channel-wise by σ and add μ to get the transformed feature map $\hat{\mathbf{f}}$, as in (7).

$$\hat{f}_{ijk} = \sigma_k \circ f_{ijk} + \mu_k \tag{7}$$

where $\sigma = M_\sigma(\lambda_0)$ and $\mu = M_\mu(\lambda_0)$, and ijk correspond to the feature location within the $W \times H \times C$ scope. Following this technique, the policy can be fully learned and conditioned with λ_0 at inference time, providing different attention ratios while maintaining accuracy. The reward-conditional training framework has been validated for the challenging application under study. However, a complete analysis and validation of the reward-conditional framework is out of the scope of this work and have been proposed for future work.

D. SYSTEM ARCHITECTURE

The architecture of the system is designed with the aim of maximizing the computation efficiency. MobileNetV1 [16] and MobileNetV2 [17] architectures have been utilized as feature extractors $\mathbf{f}$ or backbones. MobileNetV1 and MobileNetV2 backbones have been slightly modified,

to standardize their output size, with respect to the feature aggregator input size. The *conv*14 layer in MobileNetV1 has been removed. In MobileNetV2, the stride has been set to 1 in the 6-th bottleneck layer, and the number of channels has been reduced to 512 in the last *conv* layer (prior to the classification stage in the original model). For both backbones, a depth-multiplier $\alpha = 1.0$ has been selected for experimentation, which allows for a baseline model capacity to extract representative features in the context of the presented datasets. With proper model capacity, the models are less biased, and the performance of our additional adaptive policy, whose primary inputs are the stated extracted features, can be adequately validated. Nevertheless, our framework provides sufficient generality to use arbitrary feature extractor designs ($\alpha > 0$, ResNet variants [82], etc.). The receptive field size of MobileNetV1 and MobileNetV2 for the last layer in each feature extractor is 219 px $\times$ 219 px and 395 px $\times$ 395 px, respectively.

The feature aggregator **a** is composed of a ConvLSTM cell, which is in charge of aggregating both time and spatial features. The ConvLSTM cell has been implemented as in [50]. The bottleneck design of the ConvLSTM and its depth-wise separable convolutions reduce the computational costs. This fact is crucial, since the temporal aggregator has to be executed for every frame of the sequence. The size of the hidden and cell states of the ConvLSTM have been adapted based on the input image size. For the detector **d**, an SSDLite architecture [17] has been adopted, with separable convolutions and depth-wise channel size in all layers. Finally, the policy network $\pi(s_t|\theta, \lambda_0)$ has been trained with Proximal Policy Optimization (PPO) [37] algorithm, which is an actor-critic algorithm. The architecture has been shown in Table 2, without accounting the extra FCMs required for the reward-conditional training for clarity. The FCMs corresponding to the reward-conditional training are shallow fully-connected models with one hidden layer each (varying from 128 to 512 units) and one output layer (varying from 64 to 512 units). The layers corresponding to the value network, which shares weights with the policy network, have been also represented in Table 2.

TABLE 2. The proposed policy network architecture. For clarity, the layers corresponding to the reward-conditional training FCMs have not been included. The layers from conv1 to fc1 are shared layers between the policy and the value models. The fully connected layers are relative to a 320 × 320 input image resolution.

Layer	Size	Stride
conv1	3 x 3 x 1024 x 128	1
conv2	3 x 3 x 128 x 128	1
pool1	2 x 2 x 128	2
fc1	(3200 + 20 + 4) x 512	-
pi_fc1	512 x 128	-
pi_fc2	128 x 128	-
pi_fc3	128 x 128	-
pi_fc4	128 x 1	-
v_fc1	512 x 128	-
v_fc2	128 x 128	-
v_fc3	128 x 128	-
v_fc4	128 x 1	-

IV. EXPERIMENTS AND RESULTS

In the following section, the proposed inattentional framework has been thoroughly tested and validated. The training methodology section describes the staged training procedure for the presented system. The results section shows the experimentation outcome corresponding to every component of our system in a wide variety of scenarios. Finally, the discussion section interrelates the results to extract insights on our framework design.

A. TRAINING METHODOLOGY

As explained in previous sections, the proposed system is composed of several components. In this context, the training is divided into three phases: feature extractor (and detector), temporal aggregator (and detector) and adaptive inattentional policy training. Although the complete system can potentially be trained end-to-end, we approached a staged training pipeline in order to avoid convergence issues. The feature extractor, the temporal aggregator and detector have been trained in PyTorch [83], while the adaptive inattentional policy has been trained in Tensorflow [84]. The GPU used for training has been an Nvidia GeForce RTX 2080 Ti. All the models have been trained on a subset of 260,844 images from Imagenet VID 2015 dataset [53] and on 29,500 images from a custom Multirotor Aerial Vehicles VID[1] (MAV-VID) dataset, which has been made publicly available (validation sets are composed of 3,139 and 10,732 images, respectively). The Imagenet VID 2015 dataset has been reduced by selecting the 7-th smallest classes in terms of bounding boxes area (varying from 18% to 30% of the full-size image). The resulting experiments provide the generality of the public Imagenet dataset, and additionally represent the ultimate goal of the proposed framework, which is reusing context features when images context is notable across frames. Furthermore, our custom MAV-VID dataset constitutes the target application domain for the case under study, where the object to be detected appears normally small-sized in the image plane, such as the case of flying multirotor vehicles.

1) FEATURE EXTRACTOR

In the first stage, the feature extractor has been trained in conjunction with the SSD detector header, in order to extract robust features to be used by the following stages. Random single images from the video sequences and a batch of 16 images have been used. Adam has been selected as the optimizer, with a learning rate of 10^{-4}. We use an input resolution of 320 $\times$ 320 and *reduce on plateau* learning rate scheduler. We include hard negative mining and data augmentation as described in [81]. The original hard negative mining approach has been adjusted by allowing a ratio of 10 negative examples for each positive while scaling each negative loss by

[1]https://www.kaggle.com/alejodosr/multirotor-aerial-vehicle-vid-mavvid-dataset

0.3. The validation phase is carried out through the whole set of validation sequences of the datasets.

2) TEMPORAL AGGREGATOR

In the second stage, the feature extractor weights have been frozen during training. For MobileNetV1, the *conv*14 layer has been removed in order to inject the ConvLSTM layer. Again, the temporal aggregator is trained along with the SSD detection headers. We sample random sequences (of 10 frames each) from the training set for training. We unroll the LSTM to 10 frames in which we apply the same data augmentation techniques as in previous stage. The same data augmentation transformation is applied to every frame of each sequence in order to avoid missing the correlation between consecutive frames. We use a batch of 50 sequences and an image input size of 320×320.

Additionally, an extended data augmentation technique has been included to account for the structure of the synthetic feature maps. As explained in Section III-A, due to the procedure of context reuse, where a reduced feature map is placed in a context feature map, artifacts can appear close to the placement volume. In order to tackle this effect, during training, the feature map which results from the feature extraction stage is randomly modified with the ground-truth context of previous frame. We set a 0.01 probability for a feature map to be modified by this technique. Adam optimizer has been utilized with a learning rate of 10^{-4} and *reduce on plateau* learning rate scheduler. The validation is carried out through the validation sequences in each dataset.

3) ADAPTIVE INATTENTIONAL POLICY

The adaptive inattetional policy has been trained using PPO algorithm. PPO is a sample-efficient and actor-critic deep reinforcement learning algorithm. A clip range of 0.1, trajectories of 128 steps and minibatches of 4 elements have been utilized. The policy network has been trained with Adam and a learning rate of $2.5 \cdot 10^{-4}$ (refer to the code repository Supplementary Material section for remaining hyperparameters definition). The observation state has been normalized in the environment based on its mean and variance through time (at a certain number of steps, normalization parameters calculation is stopped). The reward has not been normalized, since its original design is meant for relative performance signaling. λ_0 parameter has been sampled from a log-uniform distribution every 10 sequences. Training has been carried out with the validation set of each dataset. In order to provide enough variability and avoid overfitting, a random transform as in [81] has been applied to every sequence. During testing, the random transform is not applied to allow for comparison.

B. RESULTS

For all results, the standard Imagenet VID accuracy metric is reported, mean Average Precision mAP@0.5 IOU. The latency of the networks in milliseconds (ms) and the average Frames per Second (FPS) of the complete approach is provided. The number of parameters is reported as benchmarks for efficiency. Also, the power consumption in Watts (W) and the energy efficiency in FPS/W are included for every experiment, as reported in [11]–[13].

The validation tests have been performed in two different GPUs, an Nvidia GeForce RTX 2080 Ti desktop GPU and an Nvidia Volta embedded GPU inside an Nvidia Jetson AGX Xavier platform. Also, since the latency reduction greatly stands out when the parallelization capacity is decreased, such as the case of desktop and embedded CPUs, the proposed system has been further tested and validated in an Intel Core i7-9700K@3.60GHz desktop CPU and a ARMv8.2@1.377GHz embedded CPU. All the tests have been performed in Python 3.6 and Ubuntu 18.04 operating system. The code has been made publicly available (see Supplementary Material).

In Table 3 and Table 4, the results corresponding to MobileNetV1 feature extractor for both Imagenet VID 2015 and MAV-VID dataset are shown. In addition, in Table 5 and Table 6, the results corresponding to MobileNetV2 feature extractor for both Imagenet VID 2015 and MAV-VID dataset are also depicted.

In order to further validate the performance of the proposed inattentional policy, a comparison with a random baseline for a wide variety of scenarios has been carried out. In Fig. 5 the trade-off between the number of inattentional frames executed and the resulting mAP is illustrated for both datasets and feature extractors. For every value of the reward-conditional parameter λ_0, there is an overall percentage of inattentional frames executed. In order to be able to compare it to a random baseline, a random policy has been executed the same exact amount of inattentional frames. Also, the original mAP (with no inattentional frames involved) is included. Every test has been performed 5 times each, for both policies, and the average results have been plotted.

Finally, four application cases have been illustrated in Fig. 6. Two complex scenarios, due to object high-motion speed within the image plane or environmental complexity (camera pointing to the sun); and two simple (or static) scenarios, where the object stays almost static within the image plane, have been additionally depicted.

V. DISCUSSION

The thorough testing and validation of the proposed inattentional system describes a proper framework to adaptively reduce computation overhead in the context of video object detection. A wide variety of scenarios have been tested for two feature extractors (MobileNetV1 and MobileNetV2) and two datasets (Imagenet VID 2015 and MAV-VID). Regarding Table 3, 4, 5 and 6, both architectures provide state-of-the-art mAP@0.5IOU in the Imagenet VID 2015 dataset, resulting in 0.5236 and 0.4924 for MobileNetV1 and MobileNetV2 feature extractors, respectively. Also, the mAP results in our custom MAV-VID dataset are notable, with 0.9398 and 0.9453 for MobileNetV1 and MobileNetV2 feature extractors, respectively. MobileNetV1 provided higher mAP in

TABLE 3. Results using the MobileNetV1 feature extractor on the ImageNet VID 2015 dataset. The mean Average Precision (mAP), KeyFrame (KF) and Inattentional Frame (IF) latency, as well as the average runtime FPS are provided. Also, the number of parameters of the models has been included. The FPS ratio with respect to the base FPS value (without the inattentional policy) has been also included for clarity. Additionally, power consumption and energy efficiency have been provided for every experiment.

Model	Platform	mAP	KF (ms)	IF (ms)	FPS	FPS Ratio	Params (M)	Power (W)	Efficiency (FPS/W)
MobileNetV1-SSDLite	GPU (Desktop)	0.4824	20.67	-	48.37	-	4.48	128.3	0.377
	GPU (Xavier)		84.53	-	11.83	-		4.19	2.82
	CPU (Desktop)		159.73	-	26.26	-		99.1	0.264
	CPU (Xavier)		7018.31	-	0.142	-		2.91	0.048
MobileNetV1-ConvLSTM-SSDLite	GPU (Desktop)	**0.5236**	19.33	-	51.73	-	3.3	126.8	0.407
	GPU (Xavier)		83.73	-	11.94	-		3.82	3.12
	CPU (Desktop)		39.53	-	25.29	-		98.9	0.255
	CPU (Xavier)		6378.85	-	0.156	-		2.87	0.054
MobileNetV1-ConvLSTM-SSDLite [λ_0=0.01]	GPU (Desktop)	0.5235	19.59	18.78	51.91	1.0	5.5	141.5	0.366
	GPU (Xavier)		83.33	76.63	12.4	1.03		4.25	2.91
	CPU (Desktop)		39.62	25.01	29.65	1.17		103.1	0.287
	CPU (Xavier)		6512.13	4132.13	0.1801	1.15		3.03	0.059
MobileNetV1-ConvLSTM-SSDLite [λ_0=0.5]	GPU (Desktop)	0.4964	19.78	18.64	52.67	1.0	5.5	141.5	0.372
	GPU (Xavier)		82.73	77.36	12.66	1.06		4.25	2.97
	CPU (Desktop)		39.12	25.13	34.06	1.34		103.1	0.33
	CPU (Xavier)		6532.76	4189.74	0.2041	1.3		3.03	0.067
MobileNetV1-ConvLSTM-SSDLite [λ_0=1.2]	GPU (Desktop)	0.4781	19.99	18.68	52.85	1.0	5.5	141.5	0.373
	GPU (Xavier)		83.1	75.15	13.01	1.08		4.25	3.06
	CPU (Desktop)		39.33	25.07	36.11	1.42		103.1	0.35
	CPU (Xavier)		6580.88	4063.28	0.2209	1.41		3.03	0.072
MobileNetV1-ConvLSTM-SSDLite [λ_0=2.0]	GPU (Desktop)	0.4373	19.91	18.23	**54.48**	**1.05**	5.5	141.5	**0.385**
	GPU (Xavier)		83.46	76.36	**13.15**	**1.1**		4.25	**3.09**
	CPU (Desktop)		39.89	25.45	**37.73**	**1.49**		103.1	**0.365**
	CPU (Xavier)		6593.12	4028.2	**0.2372**	**1.52**		3.03	**0.078**

TABLE 4. Results using the MobileNetV1 feature extractor on the MAV-VID dataset. The mean Average Precision (mAP), KeyFrame (KF) and Inattentional Frame (IF) latency, as well as the average runtime FPS are provided. The FPS ratio with respect to the base FPS value (without the inattentional policy) has been also included for clarity. Additionally, power consumption and energy efficiency have been provided for every experiment.

Model	Platform	mAP	KF (ms)	IF (ms)	FPS	FPS Ratio	Power (W)	Efficiency (FPS/W)
MobileNetV1-SSDLite	GPU (Desktop)	0.9183	18.85	-	53.05	-	128.2	0.413
	GPU (Xavier)		90.66	-	11.03	-	4.16	2.65
	CPU (Desktop)		122.58	-	8.15	-	99.0	0.082
	CPU (Xavier)		6406.07	-	0.156	-	2.93	0.053
MobileNetV1-ConvLSTM-SSDLite	GPU (Desktop)	**0.9398**	17.84	-	56.05	-	126.7	0.442
	GPU (Xavier)		85.54	-	11.69	-	3.79	3.08
	CPU (Desktop)		85.75	-	11.66	-	99.0	0.117
	CPU (Xavier)		5375.18	-	0.186	-	2.91	0.063
MobileNetV1-ConvLSTM-SSDLite [λ_0=0.01]	GPU (Desktop)	0.9382	17.89	16.12	55.91	0.99	141.4	0.395
	GPU (Xavier)		85.58	70.78	11.69	1.0	4.19	2.83
	CPU (Desktop)		85.81	27.62	11.68	1.0	103.0	0.116
	CPU (Xavier)		5354.33	3967.07	0.1869	1.0	3.15	0.061
MobileNetV1-ConvLSTM-SSDLite [λ_0=0.5]	GPU (Desktop)	0.9377	17.95	16.38	55.93	0.99	141.4	0.395
	GPU (Xavier)		84.77	71.2	11.88	1.01	4.19	2.78
	CPU (Desktop)		85.73	26.94	12.03	1.03	103.0	0.116
	CPU (Xavier)		5256.23	3845.12	0.1925	1.03	3.15	0.061
MobileNetV1-ConvLSTM-SSDLite [λ_0=1.2]	GPU (Desktop)	0.8845	17.96	16.39	58.41	1.04	141.4	0.413
	GPU (Xavier)		85.21	71.04	12.88	1.1	4.19	3.07
	CPU (Desktop)		85.9	27.91	18.25	1.56	103.0	0.177
	CPU (Xavier)		5312.47	3834.12	0.2212	1.18	3.15	0.07
MobileNetV1-ConvLSTM-SSDLite [λ_0=2.0]	GPU (Desktop)	0.8403	17.96	16.68	**58.86**	**1.05**	141.4	**0.416**
	GPU (Xavier)		86.58	70.99	**13.37**	**1.14**	4.19	**3.19**
	CPU (Desktop)		85.8	26.74	**24.39**	**2.09**	103.0	**0.236**
	CPU (Xavier)		5214.32	3956.01	**0.234**	**1.25**	3.15	**0.074**

Imagenet VID 2015 dataset, whereas MobileNetV2 yielded higher mAP in the MAV-VID dataset.

The mAP gets reduced when the reward-conditional parameter λ_0 increases, since the policy is encouraged to execute more inattentional frames at the cost of accuracy. Nevertheless, it maintains competitive values throughout the whole range, with a minimum accuracy of 0.4373 and 0.8403 for Imagenet VID 2015 and MAV-VID dataset, respectively. As shown in Table 5 and 6, MobileNetV2-ConvLSTM-SSDLite has remained as the model with lower mAP degradation.

The proposed inattentional framework has achieved a considerable latency reduction, increasing the runtime FPS in every platform tested, with minimal accuracy drop.

TABLE 5. Results using the MobileNetV2 feature extractor on the ImageNet VID 2015 dataset. The mean Average Precision (mAP), KeyFrame (KF) and Inattentional Frame (IF) latency, as well as the average runtime FPS are provided. Also, the number of parameters of the models has been included. The FPS ratio with respect to the base FPS value (without the inattentional policy) has been also included for clarity. Additionally, power consumption and energy efficiency have been provided for every experiment.

Model	Platform	mAP	KF (ms)	IF (ms)	FPS	FPS Ratio	Params (M)	Power (W)	Efficiency (FPS/W)
MobileNetV2-SSDLite	GPU (Desktop)	0.4775	35.79	-	27.94	-	4.84	127.3	0.219
	GPU (Xavier)		117.34	-	8.52	-		4.31	1.97
	CPU (Desktop)		297.88	-	3.35	-		98.2	0.034
	CPU (Xavier)		9001.32	-	0.1111	-		3.01	0.036
MobileNetV2-ConvLSTM-SSDLite	GPU (Desktop)	**0.4924**	36.91	-	27.09	-	3.3	128.4	0.21
	GPU (Xavier)		126.82	-	7.7	-		5.03	1.53
	CPU (Desktop)		261.62	-	3.82	-		99.1	0.038
	CPU (Xavier)		8822.3	-	0.1133	-		3.12	0.036
MobileNetV2-ConvLSTM-SSDLite [λ_0=0.01]	GPU (Desktop)	0.491	35.64	34.81	28.14	1.03	5.5	142.5	0.197
	GPU (Xavier)		126.66	118.53	7.96	1.03		5.17	1.53
	CPU (Desktop)		263.126	190.95	3.94	1.03		102.9	0.038
	CPU (Xavier)		8852.42	6267.98	0.1176	1.03		3.23	0.036
MobileNetV2-ConvLSTM-SSDLite [λ_0=0.5]	GPU (Desktop)	0.4818	36.83	34.39	28.03	1.03	5.5	142.5	0.196
	GPU (Xavier)		126.32	118.79	8.14	1.05		5.17	1.57
	CPU (Desktop)		264.02	163.84	4.62	1.2		102.9	0.044
	CPU (Xavier)		8752.36	6189.12	0.1312	1.15		3.23	0.04
MobileNetV2-ConvLSTM-SSDLite [λ_0=1.2]	GPU (Desktop)	0.4704	36.67	34.71	28.24	1.04	5.5	142.5	0.198
	GPU (Xavier)		126.08	119.1	8.22	1.06		5.17	1.58
	CPU (Desktop)		264.14	152.34	5.2	1.36		102.9	0.05
	CPU (Xavier)		8952.78	6104.78	0.1406	1.24		3.23	0.043
MobileNetV2-ConvLSTM-SSDLite [λ_0=2.0]	GPU (Desktop)	0.4622	36.08	34.19	**28.89**	**1.06**	5.5	142.5	**0.202**
	GPU (Xavier)		126.25	118.68	**8.3**	**1.07**		5.17	**1.6**
	CPU (Desktop)		264.22	144.99	**5.83**	**1.52**		102.9	**0.056**
	CPU (Xavier)		8876.23	6123.43	**0.1485**	**1.31**		3.23	**0.045**

TABLE 6. Results using the MobileNetV2 feature extractor on the MAV-VID dataset. The mean Average Precision (mAP), KeyFrame (KF) and Inattentional Frame (IF) latency, as well as the average runtime FPS are provided. The FPS ratio with respect to the base FPS value (without the inattentional policy) has been also included for clarity. Additionally, power consumption and energy efficiency have been provided for every experiment.

Model	Platform	mAP	KF (ms)	IF (ms)	FPS	FPS Ratio	Power (W)	Efficiency (FPS/W)
MobileNetV2-SSDLite	GPU (Desktop)	0.9218	25.819	-	38.73	-	127.4	0.304
	GPU (Xavier)		120.82	-	8.27	-	4.32	1.91
	CPU (Desktop)		222.92	-	4.48	-	98.1	0.045
	CPU (Xavier)		7645.68	-	0.1308	-	3.16	0.041
MobileNetV2-ConvLSTM-SSDLite	GPU (Desktop)	**0.9453**	29.78	-	33.57	-	128.5	0.261
	GPU (Xavier)		123.15	-	8.12	-	5.14	1.57
	CPU (Desktop)		230.7	-	4.33	-	99.2	0.043
	CPU (Xavier)		8002.68	-	0.125	-	3.13	0.039
MobileNetV2-ConvLSTM-SSDLite [λ_0=0.01]	GPU (Desktop)	0.9453	29.2	-	34.24	1.02	142.6	0.24
	GPU (Xavier)		123.02	-	8.12	1.0	5.11	1.58
	CPU (Desktop)		231.25	-	4.32	1.0	102.8	0.042
	CPU (Xavier)		7903.04	-	0.1265	1.0	3.13	0.04
MobileNetV2-ConvLSTM-SSDLite [λ_0=0.5]	GPU (Desktop)	0.9452	29.13	28.14	34.33	1.02	142.6	0.24
	GPU (Xavier)		122.47	115.23	8.16	1.0	5.11	1.59
	CPU (Desktop)		230.91	50.75	4.33	1.0	102.8	0.042
	CPU (Xavier)		7877.87	3319.54	0.127	1.01	3.13	0.04
MobileNetV2-ConvLSTM-SSDLite [λ_0=1.2]	GPU (Desktop)	0.86	29.08	28.47	34.78	1.03	142.6	0.243
	GPU (Xavier)		123.11	115.1	8.41	1.03	5.11	1.64
	CPU (Desktop)		232.09	51.77	7.43	1.71	102.8	0.07
	CPU (Xavier)		8009.22	3243.22	0.1842	1.47	3.13	0.058
MobileNetV2-ConvLSTM-SSDLite [λ_0=2.0]	GPU (Desktop)	0.8364	28.88	28.49	**34.93**	**1.04**	142.6	**0.244**
	GPU (Xavier)		123.15	114.65	**8.51**	**1.04**	5.11	**1.66**
	CPU (Desktop)		232.04	51.22	**8.92**	**2.06**	102.8	**0.086**
	CPU (Xavier)		7865.12	3500.33	**0.2013**	**1.61**	3.13	**0.064**

Nevertheless, the amount of computation reduction is highly dependent on the platform where the system is executed, as well as on the average object size within the image plane. Considering the GPU platforms, with a higher parallelization capacity, the average FPS increase ratio has ranged from 1.0 to 1.14 for MobileNetV1 and from 1.0 to 1.07 for MobileNetV2. However, regarding the CPU platforms, where the parallelization capacity is limited, the average FPS increase ratio has ranged from 1.0 to 2.09 for MobileNetV1 and from 1.0 to 2.06 for MobileNetV2. These results lead to an average FPS on the desktop CPU platform of 37.73, which is in the order of magnitude of the base runtime

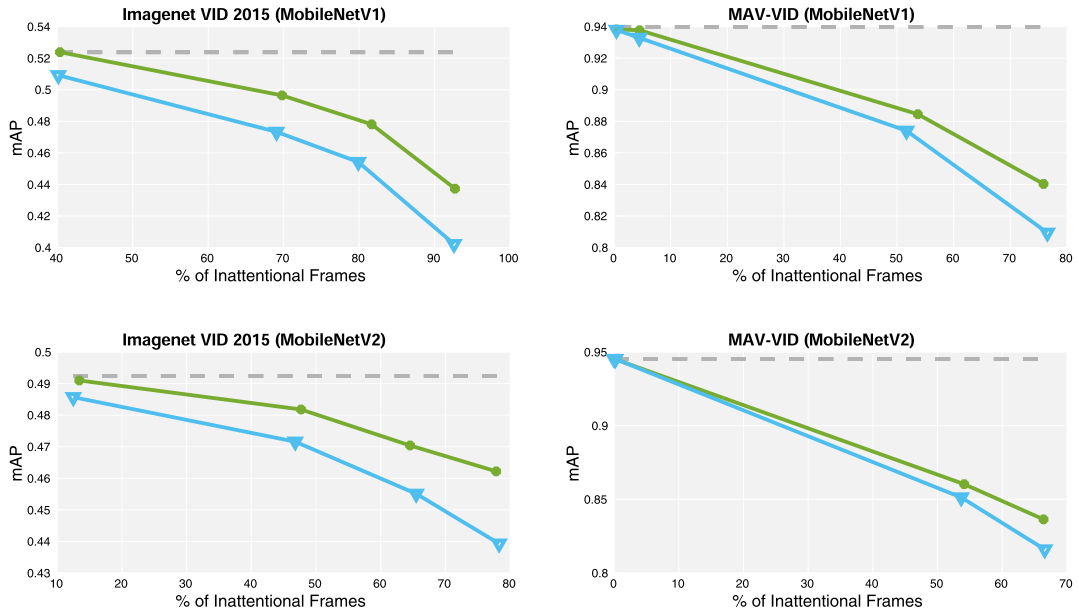

FIGURE 5. Representation of the trade-off between the mAP and the percentage of inattentional frames executed. Four cases with two feature extractors (MobileNetV1 and MobileNetV2) and two datasets (Imagenet VID 2015 and MAV-VID) have been evaluated. The random baseline policy is depicted in pale blue (circles) and the proposed inattentional policy in pale green (triangles). The original mAP (with no inattentional frames involved) is represented with a grey dashed line. The inattentional policy predominates allover the tests, resulting in a higher mAP for every percentage of inattentional frames. Also, the inattentional policy incurs less accuracy degradation even at extreme ratios of inattentional frames, suggesting that our method is superior at capturing the temporal dynamics inherent to videos. Every test has been performed 5 times each, for both policies, and the average results have been plotted.

FPS on the desktop GPU platform of 51.91, at the cost of 0.086 mAP reduction, for MobileNetV1 in Imagenet VID 2015 dataset. The maximum FPS increase ratio has been 2.09 for MobileNetV1 on the desktop CPU, achieving a runtime FPS of 24.39 at the cost of 0.09 mAP reduction, in MAV-VID dataset (where the object size is on average smaller in the image plane). These results suggest that the inattentional framework is able to increase computation efficiency when the effective parallelization capacity is limited. The effective parallelization capacity is influenced by several variables and it is relative to the input images, the model size and the platform specifications. Qualitatively, parallelization is low in average CPUs, in applications where the input images are at high resolution or in applications where the models have notably more parameters than our MobileNetV1/V2-ConvLSTM-SSDLite models. These facts yield to an open field of research, where the proposed inattentional framework can be applied. Nevertheless, such extensive study is out of the scope of the present work.

Regarding power consumption and energy efficiency, the proposed inattentional framework provides proper results in terms of both global power consumption and energy efficiency increase. Power consumption is notably higher in desktop GPU/CPUs, with on average 35× more power consumption than the embedded Xavier platform. The lowest energy consumption has been 2.87 W for MobileNetV1-ConvLSTM-SSDLite base model (no inattentional policy) in Imagenet VID 2015 dataset. Energy efficiency varies across platforms, being the GPUs more energy efficient due to their parallelization capabilities. The maximum energy efficiency has resulted in 3.19 FPS/W (1.12×) for MobileNetV1 on the Xavier GPU, achieving a runtime FPS of 13.37 at the cost of 4.19 W, in MAV-VID dataset. It has to be noted that, regarding relative energy efficiency increase with respect to the base energy efficiency (no inattentional frames), the results follows approximately the same ratios as in the FPS case, resulting in a maximum energy efficiency increase of 2.09× for MobileNetV1 on the desktop CPU, and achieving a runtime FPS of 24.39 at the cost of 0.09 mAP reduction, in MAV-VID dataset.

Considering Fig. 5, in comparison to the random baseline, our inattentional policy predominates allover the tests, providing a higher mAP for every percentage of inattentional frames executed. Furthermore, our inattentional policy shows lower mAP degradation even at extreme percentage of inattentional frames, suggesting that our method is superior at capturing the temporal dynamics inherent to videos. The highest mAP distance to the random baseline has been provided by the MobileNetV1-ConvLSTM-SSDLite in the Imagenet VID 2015 dataset, with a distance of 0.4 in mAP at 92% of inattentional frames executed. Another emergent property is that, for the case of MobileNetV1 in Imagenet VID 2015 dataset, the inattentional policy is able to match the original mAP even when executing a 40% of inattentional frames, which suggests there is a redundancy of information that does not add value to the final accuracy. In addition, thanks to our novel reward-conditional training scheme, the policy can be conditioned at inference time, providing

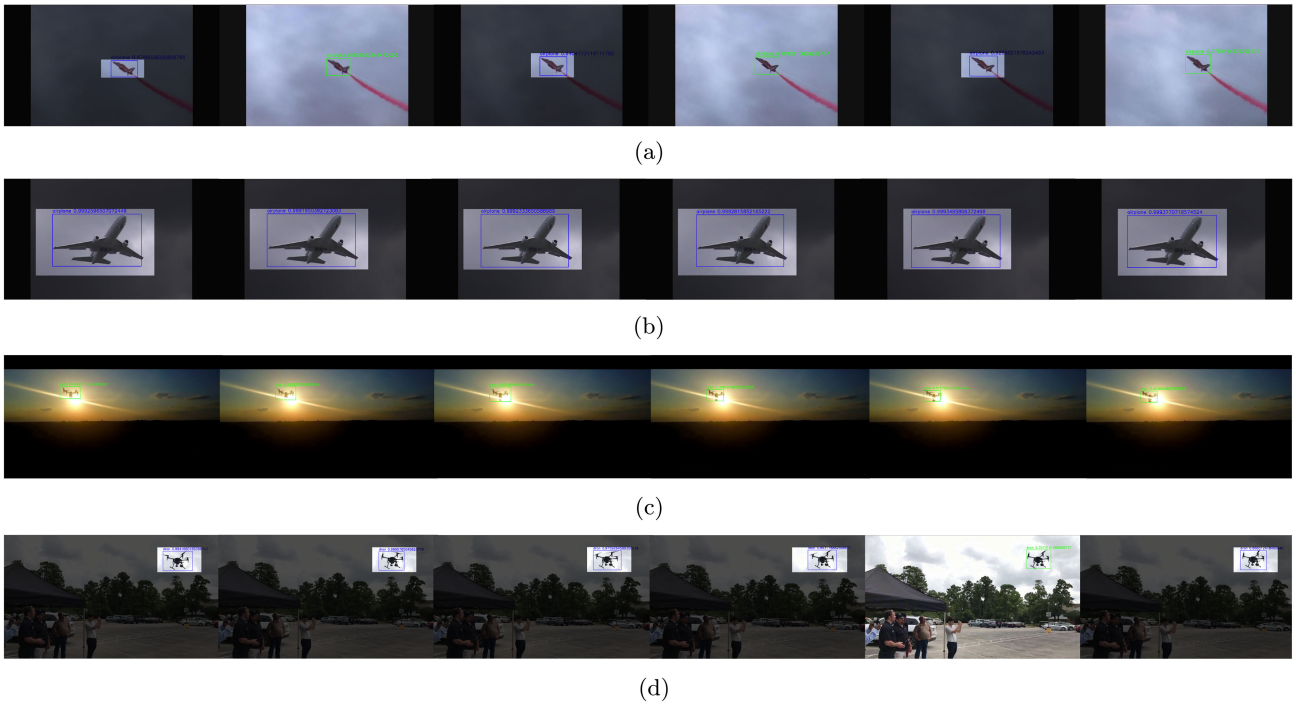

FIGURE 6. Images corresponding to four different scenarios for both datasets. Every image example has been processed by MobileNetV1-ConvLSTM-SSDLite [$\lambda_0 = 1.2$] model (MobileNetV2-ConvLSTM-SSDLite provides a similar behavior). Shaded regions correspond to non-processed pixels. (a)-(b) and (c)-(d) image examples correspond to Imagenet VID 2015 and MAV-VID dataset, respectively. (a) and (c) show complex detection examples, corresponding to an object with high-motion speed within the image plane, and to an increased environmental complexity (camera pointing to the sun), respectively. In this context, the inattentional policy executes a lower percentage of inattentional frames. (b) and (d) illustrate objects which remain more static throughout the sequence. In this scenario, the number of inattentional frames is higher.

promising results in a wide variety of sequences with varying backgrounds, changes in illumination, object size, etc. and allowing for real-time performance modulation in the context of the required application (see supplementary video[2]).

Finally, at the cost of 2.2M parameters, a learned inattentional policy can provide adaptiveness to the video dynamics, as shown in Fig. 6. In this figure, two complex and two simple scenarios, in terms of detection difficultness, are illustrated. In the complex scenarios, where the object is moving fast within the image plane, or there are ambient conditions which difficult detection, the inattentional policy performs a higher rate of full keyframes in order to maintain the detection accuracy of the object through the sequence. Nevertheless, when the scenarios are simpler, such as the case an object with slow motion in the image plane, the inattentional policy tends to neglect context to speedup computation, without missing accuracy. This adaptive behavior can be very promising in a wide variety of video object detection applications.

VI. CONCLUSION

In this work, the inattentional framework has been studied. The presented framework aims at reducing the computation overhead, in the frame of video object detection, by reusing redundant context in video images. An inattentional policy has been learned, under the reinforcement learning paradigm, to select the amount of frames where the context is reused. Furthermore, a novel reward-conditional training has been presented, where a policy can be trained on a distribution of reward functions and conditioned on one unique function at inference time. The inattentional framework provided an average latency reduction in CPUs up to 2.09 times the original latency, and obtaining FPS rates similar to their equivalent GPU platform, at the cost of a mAP reduction of 1.11 times.

This study could be extended by evaluating the inattentional framework in other scenarios of low parallelization capacity, such as the case of high resolution input images or mobile devices. Also, optimization techniques and shallower architectures can be further applied, such as half-precision inference, quantization or MobileNets with $\alpha < 1$. Regarding the reward-conditional training, a complete study of this method with a diverse set of reward functions and reinforcement learning algorithms has been left as future work.

SUPPLEMENTARY MATERIAL

The code has been made publicly available at https://github.com/alejodosr/adaptive-inattention. Also, a short video demonstration can be found at https://vimeo.com/426725929. The MAV-VID dataset can be downloaded at https://www.kaggle.com/alejodosr/multirotoraerial-vehicle-vid-mavvid-dataset.

ACKNOWLEDGMENT

The authors would like to acknowledge Estefanía Carolina Asimbaya Shuguli for the exhaustive annotations of the images for the generation of the public dataset.

[2]https://vimeo.com/426725929

REFERENCES

[1] A. Bochkovskiy, C.-Y. Wang, and H.-Y. Mark Liao, "YOLOv4: Optimal speed and accuracy of object detection," 2020, *arXiv:2004.10934*. [Online]. Available: http://arxiv.org/abs/2004.10934

[2] S. Liu, D. Huang, and Y. Wang, "Learning spatial fusion for single-shot object detection," 2019, *arXiv:1911.09516*. [Online]. Available: http://arxiv.org/abs/1911.09516

[3] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 2980–2988.

[4] M. Tan, R. Pang, and Q. V. Le, "EfficientDet: Scalable and efficient object detection," 2019, *arXiv:1911.09070*. [Online]. Available: http://arxiv.org/abs/1911.09070

[5] X. Zhou, D. Wang, and P. Krähenbühl, "Objects as points," 2019, *arXiv:1904.07850*. [Online]. Available: http://arxiv.org/abs/1904.07850

[6] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*. [Online]. Available: http://arxiv.org/abs/1804.02767

[7] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 1440–1448.

[8] R. Huang, J. Pedoeem, and C. Chen, "YOLO-LITE: A real-time object detection algorithm optimized for non-GPU computers," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2018, pp. 2503–2510.

[9] M. Gao, R. Yu, A. Li, V. I. Morariu, and L. S. Davis, "Dynamic zoom-in network for fast object detection in large images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6926–6935.

[10] A. Pirinen and C. Sminchisescu, "Deep reinforcement learning of region proposal networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6945–6954.

[11] K. Rungsuptaweekoon, V. Visoottiviseth, and R. Takano, "Evaluating the power efficiency of deep learning inference on embedded GPU systems," in *Proc. 2nd Int. Conf. Inf. Technol. (INCIT)*, Nov. 2017, pp. 1–5.

[12] H. Mao, S. Yao, T. Tang, B. Li, J. Yao, and Y. Wang, "Towards real-time object detection on embedded systems," *IEEE Trans. Emerg. Topics Comput.*, vol. 6, no. 3, pp. 417–431, Sep. 2018.

[13] J. Yu, K. Guo, Y. Hu, X. Ning, J. Qiu, H. Mao, S. Yao, T. Tang, B. Li, Y. Wang, and H. Yang, "Real-time object detection towards high power efficiency," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2018, pp. 704–708.

[14] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and < 0.5 MB model size," 2016, *arXiv:1602.07360*. [Online]. Available: http://arxiv.org/abs/1602.07360

[15] A. Howard, M. Sandler, B. Chen, W. Wang, L.-C. Chen, M. Tan, G. Chu, V. Vasudevan, Y. Zhu, R. Pang, H. Adam, and Q. Le, "Searching for MobileNetV3," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2019, pp. 1314–1324.

[16] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*. [Online]. Available: http://arxiv.org/abs/1704.04861

[17] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4510–4520.

[18] M. Tan, B. Chen, R. Pang, V. Vasudevan, and Q. V. Le, "MnasNet: Platform-aware neural architecture search for mobile," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 2815–2823.

[19] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6848–6856.

[20] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "ShuffleNet V2: Practical guidelines for efficient CNN architecture design," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 116–131.

[21] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 4107–4115.

[22] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Quantized neural networks: Training neural networks with low precision weights and activations," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 6869–6898, 2017.

[23] P. Bacchus, R. Stewart, and E. Komendantskaya, "Accuracy, training time and hardware efficiency trade-offs for quantized neural networks on FPGAs," in *Proc. Int. Symp. Appl. Reconfigurable Comput.* Toledo, Spain: Springer, 2020, pp. 121–135.

[24] C. Ding, S. Wang, N. Liu, K. Xu, Y. Wang, and Y. Liang, "REQ-YOLO: A resource-aware, efficient quantization framework for object detection on FPGAs," in *Proc. ACM/SIGDA Int. Symp. Field-Programmable Gate Arrays*, Feb. 2019, pp. 33–42.

[25] Y. Kim, M. Imani, and T. S. Rosing, "Image recognition accelerator design using in-memory processing," *IEEE Micro*, vol. 39, no. 1, pp. 17–23, Jan. 2019.

[26] J. Aloimonos, I. Weiss, and A. Bandyopadhyay, "Active vision," *Int. J. Comput. Vis.*, vol. 1, no. 4, pp. 333–356, 1988.

[27] J. M. Findlay and I. D. Gilchrist, *Active Vision: The Psychology of Looking and Seeing*, no. 37. London, U.K.: Oxford Univ. Press, 2003.

[28] H. Kolb, "Simple anatomy of the retina," in *Webvision: The Organization of the Retina and Visual System*. Salt Lake City, UT, USA: The Organization of the Retina and Visual System, 1995, pp. 13–36.

[29] H. Strasburger, I. Rentschler, and M. Jüttner, "Peripheral vision and pattern recognition: A review," *J. Vis.*, vol. 11, no. 5, p. 13, 2011.

[30] W. Kienzle, M. O. Franz, B. Schölkopf, and F. A. Wichmann, "Center-surround patterns emerge as optimal predictors for human saccade targets," *J. Vis.*, vol. 9, no. 5, p. 7, May 2009.

[31] D. Purves, G. J. Augustine, D. Fitzpatrick, W. Hall, A.-S. LaMantia, J. O. McNamara, and L. White, *Neuroscience*. Sunderland, MA, USA: Sinauer Associates, 2001.

[32] A. Mack and I. Rock, *Inattentional Blindness*. Cambridge, MA, USA: MIT Press, 1998.

[33] J. Li, F. Shi, W. Liu, D. Zou, Q. Wang, P.-K. J. Park, and H. E. Ryu, "Adaptive Temporal Pooling for Object Detection using Dynamic Vision Sensor," in *Proc. Brit. Mach. Vis. Conf. (BMVC)*, T.-K. Kim, S. Zafeiriou, G. Brostow, and K. Mikolajczyk, Eds. BMVA Press, Sep. 2017, pp. 40.1–40.12. [Online]. Available: https://dx.doi.org/10.5244/C.31.40, doi: 10.5244/C.31.40.

[34] N. F. Y. Chen, "Pseudo-labels for supervised learning on dynamic vision sensor data, applied to object detection under ego-motion," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, Jun. 2018, pp. 644–653.

[35] A. Dosovitskiy and J. Djolonga, "You only train once: Loss-conditional training of deep networks," in *Proc. Int. Conf. Learn. Represent.*, 2020.

[36] E. Perez, F. Strub, H. D. Vries, V. Dumoulin, and A. Courville, "Film: Visual reasoning with a general conditioning layer," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 3942–3951.

[37] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*. [Online]. Available: http://arxiv.org/abs/1707.06347

[38] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*. [Online]. Available: http://arxiv.org/abs/1509.02971

[39] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1889–1897.

[40] X. Zhu, Y. Xiong, J. Dai, L. Yuan, and Y. Wei, "Deep feature flow for video recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 2349–2358.

[41] X. Zhu, Y. Wang, J. Dai, L. Yuan, and Y. Wei, "Flow-guided feature aggregation for video object detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 408–417.

[42] C. Hetang, H. Qin, S. Liu, and J. Yan, "Impression network for video object detection," 2017, *arXiv:1712.05896*. [Online]. Available: http://arxiv.org/abs/1712.05896

[43] X. Zhu, J. Dai, X. Zhu, Y. Wei, and L. Yuan, "Towards high performance video object detection for mobiles," 2018, *arXiv:1804.05830*. [Online]. Available: http://arxiv.org/abs/1804.05830

[44] Z. Zhu, W. Wu, W. Zou, and J. Yan, "End-to-end flow correlation tracking with spatial-temporal attention," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 548–557.

[45] L. Zhang, Z. Lin, J. Zhang, H. Lu, and Y. He, "Fast video object segmentation via dynamic targeting network," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2019, pp. 5582–5591.

[46] S. Jain, X. Wang, and J. E. Gonzalez, "Accel: A corrective fusion network for efficient semantic segmentation on video," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 8866–8875.

[47] B. Mahasseni, S. Todorovic, and A. Fern, "Budget-aware deep semantic video segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 1029–1038.

[48] F. Xiao and Y. J. Lee, "Video object detection with an aligned spatial-temporal memory," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, pp. 485–501, 2018.

[49] M. Liu, M. Zhu, M. White, Y. Li, and D. Kalenichenko, "Looking fast and slow: Memory-guided mobile video object detection," 2019, *arXiv:1903.10172*. [Online]. Available: http://arxiv.org/abs/1903.10172
[50] M. Zhu and M. Liu, "Mobile video object detection with temporally-aware feature maps," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 5686–5695.
[51] X. Chen, J. Yu, and Z. Wu, "Temporally identity-aware SSD with attentional LSTM," *IEEE Trans. Cybern.*, vol. 50, no. 6, pp. 2674–2686, Jun. 2020.
[52] S. S. Nabavi, M. Rochan, and W. Yang, "Future semantic segmentation with convolutional LSTM," 2018, *arXiv:1807.07946*. [Online]. Available: http://arxiv.org/abs/1807.07946
[53] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.
[54] W. Han, P. Khorrami, T. Le Paine, P. Ramachandran, M. Babaeizadeh, H. Shi, J. Li, S. Yan, and T. S. Huang, "Seq-NMS for video object detection," 2016, *arXiv:1602.08465*. [Online]. Available: http://arxiv.org/abs/1602.08465
[55] K. Kang, H. Li, J. Yan, X. Zeng, B. Yang, T. Xiao, C. Zhang, Z. Wang, R. Wang, X. Wang, and W. Ouyang, "T-CNN: Tubelets with convolutional neural networks for object detection from videos," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 10, pp. 2896–2907, Oct. 2018.
[56] K. Kang, W. Ouyang, H. Li, and X. Wang, "Object detection from video tubelets with convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 817–825.
[57] H. Deng, Y. Hua, T. Song, Z. Zhang, Z. Xue, R. Ma, N. Robertson, and H. Guan, "Object guided external memory network for video object detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2019, pp. 6678–6687.
[58] L. Yang, Y. Fan, and N. Xu, "Video instance segmentation," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2019, pp. 5188–5197.
[59] H. Mao, T. Kong, and W. J. Dally, "CaTDet: Cascaded tracked detector for efficient object detection from video," 2018, *arXiv:1810.00434*. [Online]. Available: http://arxiv.org/abs/1810.00434
[60] H. Luo, W. Xie, X. Wang, and W. Zeng, "Detect or track: Towards cost-effective video object detection/tracking," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 8803–8810.
[61] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Detect to track and track to detect," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 3038–3046.
[62] Z. Jiang, P. Gao, C. Guo, Q. Zhang, S. Xiang, and C. Pan, "Video object detection with locally-weighted deformable neighbors," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 8529–8536.
[63] G. Bertasius, L. Torresani, and J. Shi, "Object detection in video with spatiotemporal sampling networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 331–346.
[64] K. Chen, J. Wang, S. Yang, X. Zhang, Y. Xiong, C. C. Loy, and D. Lin, "Optimizing video object detection via a scale-time lattice," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7814–7823.
[65] Y. Li, J. Shi, and D. Lin, "Low-latency video semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 5997–6005.
[66] C. Ying and K. Fragkiadaki, "Depth-adaptive computational policies for efficient visual tracking," in *Proc. Int. Workshop Energy Minimization Methods Comput. Vis. Pattern Recognit.* Venice, Italy: Springer, 2017, pp. 109–122.
[67] J. Supancic and D. Ramanan, "Tracking as online decision-making: Learning a policy from streaming videos with reinforcement learning," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 322–331.
[68] X. Hou and L. Zhang, "Saliency detection: A spectral residual approach," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2007, pp. 1–8.
[69] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 11, pp. 1254–1259, 1998.
[70] M. Najibi, B. Singh, and L. Davis, "AutoFocus: Efficient multi-scale inference," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2019, pp. 9745–9755.
[71] C. Chen, M.-Y. Liu, O. Tuzel, and J. Xiao, "R-CNN for small object detection," in *Proc. Asian Conf. Comput. Vis.* Taipei, Taiwan: Springer, 2016, pp. 214–230.
[72] Y. Lu, T. Javidi, and S. Lazebnik, "Adaptive object detection using adjacency and zoom prediction," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 2351–2359.
[73] S. Gidaris and N. Komodakis, "Object detection via a multi-region and semantic segmentation-aware CNN model," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 1134–1142.
[74] Y. Zhu, R. Urtasun, R. Salakhutdinov, and S. Fidler, "SegDeepM: Exploiting segmentation and context in deep neural networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 4703–4711.
[75] J. Ba, V. Mnih, and K. Kavukcuoglu, "Multiple object recognition with visual attention," 2014, *arXiv:1412.7755*. [Online]. Available: http://arxiv.org/abs/1412.7755
[76] S. Mathe, A. Pirinen, and C. Sminchisescu, "Reinforcement learning for visual object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 2894–2902.
[77] J. C. Caicedo and S. Lazebnik, "Active object localization with deep reinforcement learning," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 2488–2496.
[78] B. Uzkent and S. Ermon, "Learning when and where to zoom with deep reinforcement learning," 2020, *arXiv:2003.00425*. [Online]. Available: http://arxiv.org/abs/2003.00425
[79] B. Uzkent, C. Yeh, and S. Ermon, "Efficient object detection in large images using deep reinforcement learning," in *Proc. IEEE Winter Conf. Appl. Comput. Vis.*, Mar. 2020, pp. 1824–1833.
[80] Z. Jie, X. Liang, J. Feng, X. Jin, W. Lu, and S. Yan, "Tree-structured reinforcement learning for sequential object localization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 127–135.
[81] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.* Amsterdam, The Netherlands: Springer, 2016, pp. 21–37.
[82] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.
[83] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein L. Antiga, and A. Desmaison, "PyTorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 8024–8035.
[84] M. Abadi *et al.*, "TensorFlow: A system for large-scale machine learning," in *Proc. 12th USENIX Symp. Oper. Syst. Design Implement. (OSDI)*, 2016, pp. 265–283.

ALEJANDRO RODRIGUEZ-RAMOS (Graduate Student Member, IEEE) received the M.Sc. degree in telecommunication engineering (major in electronics and micro-electronics) from the Universidad Politécnica de Madrid (UPM), Madrid, Spain, in 2015, where he is currently pursuing the Ph.D. degree with the Centre for Automation and Robotics, Computer Vision and Aerial Robotics (CVAR) Group. H was a Visiting Researcher with the Aerospace Controls Laboratory (ACL), Massachusetts Institute of Technology (MIT), from October to December 2018, for three months. He worked for more than a year in the aerospace sector, contributing to projects from the European Space Agency (ESA). He is currently a Researcher with the Computer Vision and Aerial Robotics (CVAR) Group, Centre for Automation and Robotics, Universidad Politécnica de Madrid. His research interests include deep reinforcement learning techniques applied to aerial robotics, deep learning, aerial robotics, and image processing. He has received several international prizes in UAV competitions, such as IMAV2016, IMAV 2017, and MBZIRC 2020 (Team Leader).

JAVIER RODRIGUEZ-VAZQUEZ received the B.Sc. degree in computer engineering (double major in hardware engineering and computer science) and the M.Sc. degree in systems engineering and computing research from the Universidad de Cádiz (UCA), Cádiz, Spain, in July 2015 and March 2017, respectively. He is currently pursuing the Ph.D. degree in artificial intelligence with the Universidad Politécnica de Madrid (UPM). His research interest includes deep learning methods for solving computer vision tasks, with a special interest in object detection and image segmentation. He has received an international prize in the MBZIRC 2020 Competition.

CARLOS SAMPEDRO (Member, IEEE) received the B.Sc. degree in industrial engineering (major in industrial electronics), obtaining the best marks degree award, and the master's degree in automation and robotics from the Universidad Politécnica de Madrid (UPM), Madrid, Spain, in July 2011 and 2014, respectively, and the Ph.D. degree *(cum laude)* in automation and robotics, in December 2019. He was a Visiting Researcher with Arizona State University, AZ, USA, from September 2015 to December 2015. He is actively involved in the development of deep reinforcement learning algorithms for autonomous navigation and control of unmanned aerial vehicles (UAVs). His research interest includes the applications of learning-based techniques for solving computer vision problems, with a special interest in object detection and recognition using deep learning techniques. He received the Predoctoral Grant from the Universidad Politécnica de Madrid, in January 2017.

PASCUAL CAMPOY (Senior Member, IEEE) has been a Visiting Professor with Tongji University, Shanghai, China, and QUT, Australia. He is currently a Lecturer in control, machine learning, and computer vision. He is also a Full Professor in automation and robotics with the Universidad Politécnica de Madrid (UPM), Madrid, Spain, and a Visiting Professor with TU Delft, The Netherlands. He is leading the Centre for Automation and Robotics (CAR), Computer Vision and Aerial Robotics (CVAR) Research Group. He has been the Head Director of over 40 research and development projects, including research and development European projects, national research and development projects, and over 25 technological transfer projects directly contracted with the industry. He is the author of over 200 international scientific publications. He holds nine patents, three of them registered internationally. He has received several international prizes in unmanned aerial vehicle (UAV) competitions, such as IMAV 2012, IMAV 2013, IARC 2014, IMAV2016, IMAV 2017, and MBZIRC 2020.