Article

A Video Analytics System for Person Detection Combined with Edge Computing

Evangelos Maltezos , Panagiotis Lioupis , Aris Dadoukis, Lazaros Karagiannidis, Eleftherios Ouzounoglou , Maria Krommyda and Angelos Amditis

Institute of Communication and Computer Systems (ICCS), 15773 Zografou, Greece;
panagiotis.lioupis@iccs.gr (P.L.); aristeidis.dadoukis@iccs.gr (A.D.); lkaragiannidis@iccs.gr (L.K.);
eleftherios.ouzounoglou@iccs.gr (E.O.); maria.krommyda@iccs.gr (M.K.); a.amditis@iccs.gr (A.A.)

* Correspondence: evangelos.maltezos@iccs.gr

Abstract: Ensuring citizens' safety and security has been identified as the number one priority for city authorities when it comes to the use of smart city technologies. Automatic understanding of the scene, and the associated provision of situational awareness for emergency situations, are able to efficiently contribute to such domains. In this study, a Video Analytics Edge Computing (VAEC) system is presented that performs real-time enhanced situation awareness for person detection in a video surveillance manner that is also able to share geolocated person detection alerts and other accompanied crucial information. The VAEC system adopts state-of-the-art object detection and tracking algorithms, and it is integrated with the proposed Distribute Edge Computing Internet of Things (DECIoT) platform. The aforementioned alerts and information are able to be shared, though the DECIoT, to smart city platforms utilizing proper middleware. To verify the utility and functionality of the VAEC system, extended experiments were performed (i) in several light conditions, (ii) using several camera sensors, and (iii) in several use cases, such as installed in fixed position of a building or mounted to a car. The results highlight the potential of VAEC system to be exploited by decision-makers or city authorities, providing enhanced situational awareness.

Keywords: person detection; object detection; object tracking; smart cities; situational awareness; edge computing; computer vision; YOLOv5; terrestrial; vehicle

Citation: Maltezos, E.; Lioupis, P.; Dadoukis, A.; Karagiannidis, L.; Ouzounoglou, E.; Krommyda, M.; Amditis, A. A Video Analytics System for Person Detection Combined with Edge Computing. *Computation* **2022**, *10*, 35.
<https://doi.org/10.3390/computation10030035>

Academic Editor: Chi-Wai Chow

Received: 2 February 2022

Accepted: 24 February 2022

Published: 25 February 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

In the past two decades, various aspects relating to smart cities have been examined in line with emerging technologies and situation awareness. Researchers have referenced smart technologies and smart architectures within smart cities, referring to the numerous integrated sensory devices working together and fused through larger infrastructures, focusing on data transmission, exchange, storage, processing, and security [1,2]. Moreover, new Internet of Things (IoT) applications, including Augmented Reality (AR), Artificial Intelligence (AI), and Digital Twins (DT), bring new challenges to a such increasing research domain [3]. In general, a smart city is a city that uses digital technology to protect, connect, and enhance citizens' lives, mainly referring to six characteristics: (i) Smart economy, (ii) Smart mobility, (iii) Smart environment, (iv) Smart people, (v) Smart living, and (vi) Smart governance [4]. Within such categories, there are other sub-categories, such as Smart health, Smart safety, etc. In such a context, there are several objectives and research challenges that a smart city should be able to address, such as: (i) the data from different IoT sources/devices should be available to be easily filtered, categorized, and aggregated, (ii) the exchanged data and information should be easily visualized and securely accessible, respecting privacy, (iii) categorized, detailed, measurable, and real-time knowledge should be available at several levels, (iv) scene understanding, analytics, and decision-making systems should be exploited, (v) the city should incorporate state-of-the-art technologies

for automation and further relevant extensibility, and (vi) the city should exploit several network features and collaborative spaces.

Ensuring citizens' safety and security has been identified as the number one priority for city authorities when it comes to the use of smart city technologies. Automatic understanding of the scene, and the associated provision of situational awareness for emergency situations and civil protection, are able to efficiently contribute to such domains. Situational awareness is a critical aspect of the decision-making process in emergency response and civil protection, and it requires the availability of up-to-date information on the current situation. There are various interpretations of situational awareness [5,6]; nevertheless, the most prevalent is the one from [7], which categorizes it to the following main steps: (i) perception of the current situation, (ii) comprehension of said situation, and (iii) projection of the future condition. In our work, we focus on the first element, i.e., automated perception of the current situation. Hence, a situational awareness system is necessary to provide real-time geospatial data, emergency notifications, and rapid multimedia sharing for resource management, team collaboration, and incident management. In [8], the authors proposed a framework for shaping and studying situation awareness and smart city planning, ending up with extended urban planning practice. Among others, they highlighted the demands for a sustainable and smart planning process, which understands the dynamic and systemic nature of a smart city. Additionally, they analyzed the significance of situational awareness in complex urban ecosystems with various actors. In [9], the authors explored the potential to exploit smart city IoT capabilities to help with disaster recovery operations. In their research, they described a disaster recovery scenario in the context of smart cities, as well as the security and technical challenges related to harvesting information from smart city IoT systems, for enrichment of situational awareness. In [10], the importance of the sophisticated design solution, as a proper mechanism for controlling the level of situational awareness, was proposed in three dimensions, where situational awareness, workspace awareness, and augmented reality are considered jointly. In [11], the authors investigated the threat landscape of smart cities, survey and reveal the progress in data-driven methods for situational awareness and evaluated their effectiveness when addressing various cyber threats. A platform that integrates geospatial information with dynamic data and uses domain-specific analysis and visualization to achieve situational awareness, for smart city applications, is proposed by [12] and exploited in a display of a common operational picture (COP). The situational awareness of the proposed platform was implemented by incorporating business rules, governing key performance measurement of data changes over space and time, correlating and visualizing changes, and triggering standard or predefined procedures for event handling.

Technological advancements in situational awareness apply several state-of-the-art technologies, such as machine learning, image processing, and computer vision. In recent years, there are many approaches that have been proposed that provide object detection and tracking via cameras, especially in the person detection domain, used in several applications such as self-driving vehicles, anomaly/abnormality detection during surveillance, search and rescue, etc. In this context, video surveillance is a key application which is used in most public and private places for observation and monitoring [13]. Nowadays, intelligent video surveillance and analytics systems are used which detect, track, and gain a high-level understanding of objects without human supervision. Such intelligent video surveillance systems are used in homes, offices, hospitals, malls, parking areas, and critical areas, as well as relevant infrastructures. With the rapid development in deep machine learning and convolutional neural networks (CNNs), more powerful tools, which are able to learn semantic, high-level, deeper features, have been introduced, such as YOLOv5 [14] and DeepSort [15], to address the problems of traditional architectures [16]. In [17], a general detector was applied for terrestrial use cases focused on person detection. In their research, they modified the network parameters and structure in order to adapt it to the use case. In [18], a deep learning model has been explored in the context of person detection from overhead view, while in [19], an outlining and improvement of human tracking and

detection, enriched with laser pointing, was presented. In [20], a method for fast person detection and classification of moving objects for low-power, single-board computers was proposed. The developed algorithm used geometric parameters of an object, as well as scene-related parameters, as features for classification. The extraction and classification of such features was able to be executed by low-power IoT devices. The authors of [21] conducted, in their study, a comparison among the state-of-the-art deep learning base object detector, with the focus on person detection performance in indoor environments. In [22], a state-of-the-art object detection and classifier framework on thermal vision, with seven distinct classes for advanced driver-assistance systems, was explored and adapted. In [23], a discrimination learning online person tracker was presented with an optimized trajectory extension strategy, while [24] proposed a multi-object tracking algorithm with motion perdition capabilities focused on person tracking.

On the other hand, the development of intelligent applications in IoT has gained significant attention in recent years, especially when focused on safety and security applications in smart cities, for situational awareness. Instead of cloud computing, recent trends in IoT applications have adopted edge computing that appears to decrease latency and computational processing. Hence, edge computing is able to fully contribute to applications that continuously generate enormous amounts of data of several types, while also providing a homogeneous approach for data processing and generation of associated alerts, events, or raw information. In summary, the edge computing can be used for real-time smart city environments, enabling: (i) context-awareness, (ii) geo-distributed capabilities, (iii) low latency, and (iv) the migration of computing resources from the remote cloud to the network edge. Therefore, it is important to use edge computing devices to extract the full advantage of edge concept. With the advancement in the edge devices, few contributions are observed for video surveillance and analytics in detecting anomalies (e.g., detection of objects in prohibited areas or specific zones) at the edge [16]. Since edge devices suffer from a limitation of computational resources, it is necessary to develop light weight deep learning models with high inference speed and CNN architectures. In general, the lighter the architecture, the better the performance of the edge device gets. In [25], the edge device was responsible for the most of the computational parts and hence, was faster, and when there was need for further assistance, cloud is reached. In [26], the feasibility of processing surveillance video streaming at the network edge, for real-time moving human objects tracking, was investigated via shallow traditional schemes such Histogram of Oriented Gradients (HOG) and linear Support Vector Machine (SVM) [27,28]. In [29,30], the authors presented a model for scalable object detection, tracking, and pattern recognition of moving objects that rely on dimensionality reduction with edge computing architecture. The goal of their proposed model was to build a system that is able to process large numbers of video files stored in a database or real-time videos.

Our Contribution

As mentioned above, person detection is an essential step in activity recognition, behavior analysis, and motion flow in individuals or crowd, data fusion, and modeling. However, due to the recent rise and deployment of edge computing in real-world applications, there are few contributions in the bibliography that integrate edge computing with video analytics for person detection under a smart city concept. The proposed Video Analytics Edge Computing (VAEC) system performs enhanced situation awareness for person detection in a video surveillance manner. Moreover, the VAEC is able to share, in real-time, geolocated person detection alerts from a smart city's custom data model to a middleware of other systems or platforms (e.g., smart city platform), as well as provide video frames (in which the detection is superimposed) from the video stream associated with this alert. In this context, the VAEC has a great potential to be exploited by decision-makers or city authorities under a common operational picture (COP) aspect. Summarizing, the main contributions of our work are as follows:

- A real-time and flexible system is proposed that provides enhanced situation awareness for person detection through a video streaming feed on an embedded edge device. The VAEC system is able to be placed either at indoor/outdoor premises (e.g., buildings or critical infrastructures) or mounted on a vehicle (e.g., car). Additionally, VAEC system adopts state-of-the-art algorithms responsible for object detection or a tracking-by-detection (TBD) processes (i.e., pre-trained YOLOv5 and DeepSort, respectively), utilizing several types of camera sensors (RGB or thermal). It should be noted that, depending on the use case, the VAEC system should be pre-configured to execute an object detection process or a TBD process.
- The VAEC system is integrated with the proposed edge computing platform—namely, Distribute Edge Computing Internet of Things (DECIoT). The DECIoT is a scalable, secure, flexible, fully controlled, potentially interoperable, and modular open-source framework that ensures information sharing with other platforms or systems. Through the DECIoT, the computation, data storage, and information sharing are performed together, directly in the edge device, in a real-time manner.
- Once a person (or persons) is detected, crucial information (alert with timestamp, geolocation, number of the detected persons, and relevant video frames with the detections) is provided and is able to be shared with other platforms.

Although, in this study, the VAEC is used for a use case of abnormal detection during surveillance, i.e., detection of a person in a prohibited area, the system is potentially extendable by activating multiple classes of other objects of interests (e.g., vehicles, etc.), as well as fully modular to be adapted to other applications, e.g., search and rescue, etc. To verify the utility and functionality of the VAEC system, several experiments were performed (i) in several light conditions, (ii) using several types of camera sensors, and (iii) in several viewing perspectives.

2. Materials and Methods

2.1. Proposed Methodology

In Figure 1, we present the architectural design of the VAEC system. The VAEC system consists of three main pillars: (i) the hardware components (described in detail in Section 2.2) such the edge device, the GPS device, and the camera sensor, (ii) the person detection software, based on state-of-the-art algorithms, for object detection or TBD (described in detail in Section 2.3), and (iii) the edge computing platform—that is, the DECIoT platform (described in detail in Section 2.4).

Figure 1. VAEC Architectural design.

The GPS device receiver provides, in real-time, the geolocation (latitude and longitude coordinates) of the edge device, while the camera sensor provides a video stream. The embedded edge device with GPU processing receives the information from the GPS device receiver and camera sensor. Then, the edge device executes, in real-time on the video stream, object detection or TBD algorithms to detect and count persons. More specifically, each video frame is passed to the pre-trained YOLOv5 detector (by reading the relevant deep learning weights). Then, the detector divides the image into a grid system, and each grid detects objects within itself. After the detection process is complete, a bounding box is superimposed to the video frame, showing the image plane the location of the detected objects and the relevant detection probability percentages. From the detection process, it

is able to gather some information, such the number of the detections, the timestamp of the detection, and the class of the detected object. It is mentioned that, since this study is focused on the person detection, only the class “person” of the pre-trained YOLOv5 is activated. However, the VAEC system is able to activate multiple classes (e.g., vehicles, etc.). After the detection is made, there is an optional step for a TBD process. In this context, the DeepSort tracker is used to assign an id tracking number to the detected object. A new id tracking number is assigned to every new detected object. The tracking process is completed by calculating the flow of movement of the bounding box between consecutive video frames. As a result, if an object is detected in multiple consecutive frames (in this study, three were selected), the tracker assigns an id tracking number to the detected object. The video frames of the detected person are stored as an image (overlaid with the associated bounding box) in the edge device, which has a File Transfer Protocol (FTP) Server deployed (able to share those images). Finally, the crucial information acquired from the previous steps (edge device’s geolocation, person detection alert with timestamp, number of the detected persons, and relevant folder path in the FTP server of the stored video frames with the detections) is aggregated to the DECIoT. Then, also through the DECIoT, this information is transformed, according to the smart city’s custom data model, to potentially be shared with other platforms (e.g., smart city platforms) through proper middleware. In this study, the Apache Kafka [31] was used as a middleware, considering that it represents the middleware of a smart city platform. The Apache Kafka is an effective and open-source distributed event streaming platform for high-performance data pipelines, streaming analytics, data integration, and mission-critical applications in real-world conditions, such as smart cities [32]. More details about DECIoT’s process and representative experiments are provided in Sections 2.4 and 2.5.

It should be noted that operators of VAEC system who register and/or process the images, videos, geolocation, object detection, and TBD, as well as any other data related to an identified or identifiable person, are subject to General Data Protection Regulation (GDPR), EU Data Protection Directive 95/46/EC. It is mentioned that the storage of the video frames with the detections, via the VAEC system, is optional and is able to be deactivated to further consider a privacy-oriented framework [33]. Moreover, informed consent was obtained from all subjects involved in the study.

2.2. Hardware Components

The VAEC system consists of the several hardware components that are described below (see also Figure 2):

- Camera sensor: The camera sensor provides the video stream. In this study, four camera sensors were used for the experiments. Relevant details, such as video resolution, dimensions (Height-H, Length-L, Width-W), frames per second (FPS), connectivity with the edge device, and spectrum for each camera sensor, are quoted in Table 1.
- NVIDIA Jetson TX2 (VAEC’s edge device): The Jetson TX2 is considered as an efficient and flexible solution to be as an embedded edge device with GPU processing capability [16,34]. The physical dimensions of Jetson TX2 are 18 cm × 18 cm × 5 cm. Thus, in this study, the Jetson TX2 was used as an edge device, so (i) the DECIoT is installed, (ii) has WiFi and Ethernet network connections, (iii) has a FTP server deployed, (iv) can process the video stream from the connected camera sensor, (v) executes the object detection or TBD algorithms on the video stream, (vi) stores the video frames (in which the detections are superimposed) as images, and (vii) is able to publish the alert information to other platforms or systems.
- Display screen: A display of a 7 inch touch screen LCD was used to provide autonomy, ability to control, and real-time monitoring of the object detection or TBD algorithms on the video stream. It is mentioned that the display screen is optional, and thus, it is able to be disconnected/deactivated to further consider a privacy-oriented framework [33].
- Power supply: The power supply needs of the Jetson TX2 are: 19 V, 4.7 A, and 90 W max.

- GPS device receiver: The GPS receiver provides, in real-time, the geolocation in WGS'84 coordinate system (i.e., latitude, longitude) of the VAEC system. The GPS receiver is connected with the Jetson TX2.

Table 1. Main characteristics of the camera sensors used in this study.

Camera Sensor ID	Type	Resolution	Dimensions	FPS	Spectrum	Connectivity
CAM1	Logitech BRIO 4 K Ultra HD
	4096 × 2160	H: 6.5 cm W: 10 cm L: 4.6 cm	30	RGB	USB
CAM2	Dahua IPC-HFW3841T-ZAS
	3840 × 2160	H: 7.7 cm W: 7.7 cm L: 24.5 cm	30	RGB	Ethernet
CAM3	IMX258
	1920 × 1080	H: 3.8 cm W: 3.5 cm L: 1.7 cm	30	RGB	USB
CAM4	FLIR Tau 2
	640 × 512	H: 4.5 cm W: 4.5 cm L: 7 cm	30	Thermal	USB

The specifications of the Jetson TX2 are the following: (i) RAM 4 GB 128-bit LPDDR4, (ii) AI Performance = 1.33 TFLOPS, (iii) GPU of NVIDIA Pascal architecture with 256 NVIDIA CUDA cores, and (iv) CPU = Dual-core NVIDIA Denver 2 64-bit CPU and quad-core Arm Cortex-A57 MPCore processor complex. Concerning the dependencies on networking, the networking is required when the VAEC system shares information to other systems or platforms. For the experiments of Section 3.3, a 4 G connection (for the moving car case) and a wireless network (for the building case) were required.

Figure 2. Hardware components of the VAEC's system.

2.3. Object Detection and Tracking-By-Detection (TBD)

In the bibliography, there are various robust and efficient published deep learning based object detection frameworks, which include: (i) YOLO in several versions [35–37], (ii) Single Shot MultiBox Detector (SSD) [38], (iii) R-CNN [39], (iv) Fast R-CNN [40], and (v) Mask R-CNN [41]. All of these networks are built using an end-to-end deep learning network. The efficiency of these algorithms is mostly tested with RGB datasets, which include MS COCO dataset [42], ImageNet [43], and PASCAL-VOC [44].

On the other hand, there are several efficient studies that have been proposed in the bibliography concerning object tracking algorithms. Owing to the rapid development of object detection domain, the TBD paradigm has shown effective performance. The TBD paradigm consists of a detector and a data association procedure. Initially, the detector is used to locate all objects of interest from the video sequence. Then, the feature information of each object is extracted in the data association process, and the same objects are associated according to the metrics (e.g., appearance feature and motion feature) defined on the feature. As a final step, by associating the same object in different video frames, a continuously updated tracklet set is formed. Some of the well-known and efficient trackers are: (i) MedianFlow [45], (ii) Boosting [46], (iii) GoTURN [47], (iv) Mosse [48], (v) Channel and Spatial Reliability Tracking-CSRT [49], (vi) Tracking–Learning–Detection-TLD [50], (vii) Kernelized Correlation Filter-KCF [51], (viii) Multiple instance learning-MIL [52], (ix) FairMOT [53], and (x) DeepSort [15]. It should be noted that, in TBD paradigm, the performance of the detector and the data association algorithm affect the tracking accuracy and robustness. In any case, both object detection and object tracking algorithms are mainly influenced by the sensor cameras used (e.g., spectrum capabilities, pixel resolution, etc.), viewing perspective of the objects of interest, and complexity of the scene, such as occlusions, shadows, light conditions, and background.

This study exploits the latest pre-trained version of YOLO—that is, the YOLOv5—which is responsible for the object detection process. We chose to exploit YOLOv5 for the following reasons: (i) according to our knowledge, its efficiency has not been extensively investigated in real-world applications; moreover, despite the development of effective algorithms and deep learning schemes in previous years [54], YOLOv5 seems that it has a great potential [55–57], (ii) it is freely available for deployment [14], (iii) its pre-trained weights are freely available [14] and trained by the well-known COCO dataset that contains 80 classes and more than 2,000,000 labeled images, and (iv) provides several models, such as YOLOv5s (small version), YOLOv5m (medium version), YOLOv5l (large

version), and YOLOv5x (extra-large version), which differ by the width and depth of the BottleneckCSP [58]. Since, in this study, an edge device is utilized (with limitation of computational resources) the light weight version, YOLOv5s, was exploited. The value of the mean Average Precision (mAP) for YOLOv5s is 37.2 and 56.0 with Intersection over Union (IoU) thresholds at 0.5:0.95 and 0.5, respectively [14]. According to [14], as well as from the results of the experiments in Section 3.3, the YOLOv5s satisfies the accuracy vs. computational/inference time tradeoff. Recently, an even newer version—namely, YOLOv5n (nano version)—has been released. The performance of the YOLOv5n will be investigated in the VAEC system in future work.

Since this study is focused on the person detection, only the class “person” of the pre-trained YOLOv5s weights was activated exploiting its free available version. However, the VAEC system is potentially able to use custom weights for the YOLOv5 deep learning model. In this context, a new training process, by using a transfer learning scheme [59], is required. By these means, the relevant parameters of the new deep learning model are adjusted only for the person detection (achieving maybe higher accuracy) or for only a subset of expected objects that need to be detected in the scenario. The performance of such custom weights of the YOLOv5 deep learning model will be investigated in future work.

The DeepSort tracker was also exploited for the tracking process (once a person detection occurs)—that is, combined with the YOLOv5s—in TBD manner. It should be noted that the DeepSort tracker does not implement a re-identification process. Thus, the id tracking number of a tracked person will be remaining the same till the tracker failures. If the tracker fails (e.g., due to algorithm, occlusion, color appearance and illumination, background environment, etc.), a new id tracking number is assigned to the new detected person. A re-identification process (through which lost objects are able to be re-detected and re-tracked), and a fusion of spatio-temporal and visual information, to improve re-identification accuracy will be considered in the VAEC system in future work. Furthermore, the object detection process was based on [14], while the TBD process was based on [60]. The python programming language (version 3.6) and the libraries, TensorFlow [61] and OpenCV [62], were mainly used for the object detection and TBD processes.

2.4. Distributed Edge Computing Platform (DECIoT)

Generally, edge computing refers to the enabling technology that allows computation to be performed at the network edge so that computing happens near data sources, or directly in the real-world application, as an end device. Such devices request services and information from the cloud, as well as perform several real-time computing tasks (e.g., storage, caching, filtering, processing, etc.) of the data sent to and from the cloud. The designed edge computing framework should be efficient, reliable, and secure, while also having extensibility. In this study, a novel edge computing platform called DECIoT (Figure 3) is designed and developed in order to be integrated into the VAEC system. The DECIoT platform is able to address, among others, the problem of gathering, filtering, and aggregating data, as well as interact with the IoT devices, provide security and system management, provide alerts and notifications, execute commands, store data temporarily for local persistence, transform/process data, and in the end, export the data in formats and structures that meet the needs of other platforms. This whole process is being done by using open source microservices that are state-of-the-art in the area of distributed edge IoT solutions.

The DECIoT is based on the EdgeX foundry open-source framework [63–65]. The EdgeX foundry is considered, in the bibliography, as a highly flexible and scalable edge computing framework, facilitating the interoperability between devices and applications at the IoT edge, such as industries, laboratories, and datacenters [66–69]. According to several organizations (e.g., Alliance for Internet of Things Innovation-AIOTI) [70], it is recognized, among others, as one of the Open Source Software (OSS) initiatives that are currently focusing on edge computing.

Figure 3. DECIoT architecture.

The DECIoT platform follows the microservice architecture pattern (and not the traditional monolithic architecture pattern). The main principle of the microservice architecture is that an application can be designed as a collection of loosely-coupled services, and each service is a self-contained software that is responsible for implementing a specific functionality of the application. The DECIoT platform consists of multiple layers, and each layer contains multiple microservices. The communications between the microservices, within the same or different layers, can be done either directly, with the use of REST APIs, or with the use of a message bus that follows a pub/sub mechanism. Both of them are being exploited in the VAEC system. DECIoT consists of a collection of reference implementation services and SDK tools. The micro services and SDKs are written in Go [71] or C programming languages. In the following, we present the different layers of DECIoT from southbound physical devices to northbound external middleware/infrastructures and applications, together with the explanation of which microservices were used, in the case of the VAEC system:

- The Device Service Layer acts as an interface of the system with physical devices and is tasked with the functionality of collecting data and actuating the devices with

commands. It supports multiple protocols for communication through a set of device services (MQTT Device Service, REST Device Service, Virtual Device Service, etc.) and an SDK for creating new Device Services. In the VAEC system, the MQTT Device Service was used to receive information from the object detection process. Between the object detection process and the MQTT Device Service, there is a MQTT broker (Mosquito) [72].

- The Core Services Layer is at the center of the DECIoT platform and is used for storing data, as well as commanding and registering devices. The Core Data Service is used for storing data, the Command Service initiates all the actuating commands to devices, and the Metadata Service stores all the details for the registered devices. These microservices are implemented with the use of Consul, Redis, and adapters developed in Go for integration with all other microservices. In the VAEC system, all of the above microservices have been exploited.
- The Support Services Layer includes microservices for local/edge analytics and typical application duties such as logging, scheduling, and data filtering. The Scheduling Service is a microservice capable of running periodic tasks within DECIoT (for example, cleaning the database of the Core Data Service each day) and also capable of initiating periodic actuation commands to devices using the Command Core Service. This is an implementation in Go that exploits features of Consul and Redis. The Rules Engine Service performs data filtering and basic edge data analytics, and Kuiper is used in this microservice. The Logging Service, a Go language implementation, is used for logging messages of other microservices. In this study, these microservices were not exploited in the VAEC system as no logging, scheduling, and data filtering were needed.
- The Application Services Layer consists of one or more microservices with the functionality of communicating with external infrastructures and applications. Application Services are the means to extract, transform, and send data from the DECIoT to other endpoints or Applications. Using this Layer, the DECIoT can communicate with a variety of middleware brokers (MQTT, Apache Kafka, etc.) or REST APIs with the goal of reaching external applications and infrastructures. At the same time, the support of an SDK allows the implementation of new Application Services that fit the use case. For the VAEC system, a new Application Service has been implemented to send data to the smart city's middleware (in this study, the Apache Kafka was used) using Go language.

2.5. Person Detection Alert from DECIoT to Other Platforms

The VAEC system is able to share, through the DECIoT, crucial information extracted from the object detection or TBD processes, according to smart city's custom data model, to a middleware of other systems or platforms. As mentioned above, the Apache Kafka was used as a middleware, considering that it represents the middleware of a smart city platform. In this section, the verification and demonstration of the integrated edge device with the DECIoT is presented by sending crucial information, associated with a person detection alert, to Apache Kafka associated with EXP6 (see Section 3). To verify that the Apache Kafka receives the published information from DECIoT, an Apache Kafka consumer has been executed in another external device.

To ensure that the DECIoT platform can support real time streaming of data, we measured the system latency considering the time an event has been received on MQTT as "start time" and the time a consumer consumes the event from the Apache Kafka as "end time". One hundred events were sent to DECIoT each time we executed the experiment, and the experiment has been executed 10 times, with one hour between successive runs of the experiment. The experiment was conducted within the same physical machine to avoid measuring latencies caused by networking issues. The results of this experiment were very encouraging since, on average, 31 ms latency has been measured. In the best case, a 5 ms latency has been measured, and in the worst case, 112 ms latency has been measured.

A sequence diagram that depicts the data flow is presented in Figure 4. Once the person was detected through the object detection process, the relevant message “People Detected” was generated and sent, with other crucial information, to the Mosquito MQTT broker. Then, the Device Service of the DECIoT receives the aforementioned information and passes it to the Core Data service. As next step, the Application Service receives the information from the message bus of DECIoT, transforms it as *. JSON format, and pushes it to Apache Kafka. Figure 5 shows the crucial information associated with the person detection alert, sent from the Application Service to Apache Kafka, where: (i) “source” is a description of the system/sensor that produces the data, (ii) “event_id” is the unique identification of the event, (iii) “time” is the date and timestamp for the creation of the event, (iv) “localization” is the GPS coordinates, (v) “event_type” is the type of the event, and (vi) “metadata” is the data that further describe the event, while “FTP_path” is the relevant folder path in the FTP server of the stored video frame with the detection (as an example, stored video frames are the images of EXP6 of Section 3).

Figure 4. Sequence diagram that depicts the data flow.

```

{
  "source": "VAEC",
  "event_id": "cd0b65ec-ae16-4e0b-9ffc-a4ccb92163c0",
  "time": "2022-01-17T16:50:16.75+02:00",
  "localization": {
    "lat": 37.977848,
    "lon": 23.779025,
    "elevation": 10
  },
  "event_type": "VAEC.jdl_1.alert.people_detected",
  "metadata": "ICCS, VAEC, People Detected: 1 ,Frame: 7", FTP_PATH: ftp://10.147.18.207/People_Detections/20220117/16_50_16_7.jpg"
}
  
```

Figure 5. Crucial information associated with the person detection alert sent from the Application Service of DECIoT to Apache Kafka.

3. Results

3.1. Preparation and Experimental Parameters Setting

To verify the utility and functionality of the VAEC system, several real-time experiments were performed (i) in several light conditions, (i.e., day and night), (ii) using several types of camera sensors (i.e., RGB and thermal), and in several viewing perspectives. Two viewing perspectives were considered: (i) the VAEC's edge device is placed, for surveillance purposes, to a critical building of 10 m height, with the camera looking in outdoor view, and (ii) the VAEC's edge device is mounted on a moving surveillance car. The experiments were carried out under the assumption that no person's presence is permitted to the prohibited area of interest. Thus, once a person (or persons) is detected (either for the building case or for the moving surveillance car case), it is considered as an abnormal detection. The human-in-the-loop operators that exploit the system are considered as authorized personnel under the surveillance aspect.

Concerning the battery power constraints, for the building case, the VAEC was connected, in a standard power connection manner, to a power outlet to ensure 24 h operation. For the moving car case, a power bank of 1000 Wh was used. Such a power bank is able to ensure 12 h autonomy for extensive surveillance operation. Table 2 presents the experimental condition combinations. Totally, six experiments (EXP1 to EXP6) were implemented in day light conditions with the available RGB cameras (CAM1, CAM2, and CAM3), while two experiments (EXP7 and EXP8) were conducted with the available thermal camera (CAM4). The TBD process was not used in the moving car case to avoid consecutive and useless assignments of new id tracking numbers of the detected person. Hence, only the object detection process was used for the moving car case. Due to the small size of CAM1, CAM3, and CAM4, such cameras were exploited for the moving car case.

Table 2. Considered experimental condition combinations.

Experiment ID	Camera	Light Condition	Building Case	Moving Car Case	Object Detection	TBD	Length of the Recorded Frame Sequences
EXP1	CAM1	day	✓	-	✓	-	495
EXP2	CAM1	day	✓	-	-	✓	250
EXP3	CAM1	day	-	✓	✓	-	300
EXP4	CAM2	day	✓	-	✓	-	285
EXP5	CAM2	day	✓	-	-	✓	180
EXP6	CAM3	day	-	✓	✓	-	450
EXP7	CAM4	night	✓	-	✓	-	435
EXP8	CAM4	night	-	✓	✓	-	480

On the contrary, since CAM2 is mainly designed for installation in fixed position locations, this camera was exploited only in the building case. Respectively, the CAM3 is mainly designed to be installed in moving vehicles, as it embodies a Micro Gimbal Stabilizer (MGS) and Optical Image Stabilization (OIS) feature to absorb possible turbidity and double object patterns due to the motion situation. Thus, the CAM3 was exploited only in the moving car case. For the needs of the experiments, as well as for the protection of the edge device, a relevant packaging was conducted (Figure 6). The case cover of the edge device was printed via a 3D printer.

Concerning the CAM2, a delay of the video feed (and as time was passing, the delay increased) to the edge device was observed. This is due to fact that Jetson TX2 did not have enough power to process the video frames as fast as the CAM2 was able to send them. Hence, to fix this issue, we adapted a piece of code in the python programming language, which makes a thread and captures camera frames whenever the Jetson TX2 requests them. This resulted in a small delay of maximum 1 s, which is considered acceptable compared to the constantly increasing delay that was initially observed.

Figure 6. Packaging of the edge device for the experiments.

To optimize the performance of the object detection process through YOLOv5s, the probability percentage thresholds associated with the detected person were selected as 25% (for the CAM1, CAM2, and CAM3) and 55% (for the CAM4). To avoid multiple shares and storage of duplicated information, the person detection alert and the accompanied video frames, are stored as images (in which the detections are superimposed), are provided once the number of the detected persons is updated. Additionally, to avoid incomplete and crucial information loss for detected persons, which just appear in the scene, the VAEC system verifies a detected person as detected whether it is uninterruptedly detected in three continuously video frames. Such setting can be easily modified and parameterized according to use case. Moreover, concerning the person detection alert, the corresponding alert is able to be sent to a middleware of other systems or platforms through 4 G connection (for the case the moving car) or wireless network (for the case of the building).

3.2. Evaluation Metrics

For the quantitative assessment of the object detection process, four objective criteria were adopted according to the International Society for Photogrammetry and Remote Sensing (ISPRS) guidelines [73,74]—namely, completeness (C_M), correctness (C_R), quality (Q), and F1 score measures per object (person), given as:

$$C_M = \frac{\|TP\|}{\|TP\| + \|FN\|}, C_R = \frac{\|TP\|}{\|TP\| + \|FP\|}, Q = \frac{\|TP\|}{\|TP\| + \|FP\| + \|FN\|}, F1 = 2 \times \frac{C_R \times C_M}{C_R + C_M} \quad (1)$$

where TP, FP, and FN denote true positives, false positives, and false negatives, respectively. The TP entries are the persons that exist in the scene and thus, were correctly detected. The FP entries are the persons that do not exist in scene and thus, were incorrectly detected. The FN entries are the persons that exist in the scene but were not detected.

For the quantitative assessment of the TBD, the metric Identifier Change (IDC) was adopted in order to report the number of identity swaps during the TBD process [75]. The IDC is the number of times the identifier changes for each System Track (ST), while the Ground Truth (GT) identifier is unchanged.

3.3. Experiments

Table 3 depicts representative consecutive video frames for each experiment associated with the object detection and TBD results (colored bounding boxes superimposed to the video frames). Indicative quantitative assessments for the object detection (EXP1, EXP3, EXP4, EXP6, EXP7, EXP8) and TBD (EXP2, EXP5) processes are provided in Table 4. The achieved results of the YOLOv5s (average values of $C_M = 82.8\%$, $C_R = 91.7\%$, $Q = 76.4\%$, and $F1 = 86.4\%$) are considered to be satisfactory, proving its suitability and efficiency for such applications with several challenges. For the building case, the maximum distance between the camera and the object associated with an effective and stable person detection was 45 m. On the other hand, for the moving car case, the maximum distance between the camera and the object associated with an effective and stable person detection was 45 m for the CAM1 and CAM4, and it was 35 m for the CAM3. Although CAM3 has a wide view of the scene (due to its wide angle lens), the objects are depicted smaller in the

video frames compared to the corresponding ones of CAM1 and CAM4, decreasing its performance at larger distances. The observed FN entries were mainly due to partial or total occlusions (e.g., from dense and high vegetation), which led to the reduction in the C_M rate. A representative example can be found in the bottom left image of the EXP2 row of Table 3. On the other hand, the observed FP entries were mainly due to the artificial patterns or objects with similar patterns with persons, which led to the reduction in the C_R rate. The observed FN entries were mainly due to the large distance between the camera and person (resulting in a small sized object that is unrecognizable by the algorithm) or with similar spectral values between background and the person. Figure 7 shows representative examples of FP and FN entries.

Table 3. Results of object detection and TBD processes per experiment.

Experiment ID	Representative Consecutive Video Frames					
EXP1	(a)	(b)	(c)	(d)	(e)	(f)

	(a)	(b)	(c)	(d)	(e)	(f)

	(a)	(b)	(c)	(d)	(e)	(f)

Table 3. Cont.

Experiment ID	Representative Consecutive Video Frames					
EXP3	(a)
	(b)
	(c)
	(d)
	(e)
	(f)
EXP4	(a)
	(b)
	(c)
	(d)
	(e)
	(f)
EXP5	(a)
	(b)
	(c)
	(d)
	(e)
	(f)

Table 3. *Cont.*

Experiment ID	Representative Consecutive Video Frames					
EXP6	(a)	(b)	(c)	(d)	(e)	(f)

EXP7	(a)	(b)	(c)	(d)	(e)	(f)

EXP8	(a)	(b)	(c)	(d)	(e)	(f)

Table 4. Quantitative object detection and TBD results per experiment for representative videos frames.

Object Detection Process				
Experiment ID	C _M (%)	C _R (%)	Q (%)	F1 Score
EXP1	86.4	95.0	82.6	90.5
EXP3	85.7	81.8	72.0	83.7
EXP4	87.5	95.5	84.0	91.3
EXP6	87.6	77.8	70.0	82.4
EXP7	66.7	100.0	66.7	80.0
EXP8	83.3	100.0	83.3	90.9
Average	82.8	91.7	76.4	86.4

TBD process	
Experiment id	IDC
EXP2	2
EXP5	13
Average	15

Figure 7. Observed weaknesses of the pre-trained YOLOv5s during the person detection process. Top row: FP entries due to the artificial patterns, such shadows or objects with similar patterns with persons, such as the painted pillar or graffiti; Bottom row: FN entries (magenta ellipse with dashed line) due to the large distance between the camera and person (resulting in a small sized object that is unrecognizable by the algorithm) or with similar RGB (EXP6) or thermal spectral values (EXP7 and EXP8) between background and the person.

During the produced real-time experiments, the computation time for the inference process for the detection of one person was 70 ms and 130 ms for the object detection and TBD processes, respectively. Using the same edge device, the computation time of the object detection remains the same (70 ms), even if more persons are detected. For the case of TBD, the computation time is increased when multiple persons are tracked (i.e., 300 ms

for 20 tracked persons). To ensure a low computation time, a capable edge device, with sufficient power capacity and GPU processing, is required.

Concerning the TBD process, the achieved results of the DeepSort tracker are generally considered to be satisfactory, proving its stability and effectiveness. The id tracking number of the tracked person of EXP2 and EXP5 successfully remained the same. However, for cases with occlusions, the tracker lost the tracked person, leading to the change of the corresponding id tracking number after the associated detection. Representative examples can be found in the row of EXP2 of Table 3, where the tracker loses the person twice when he walked behind the vegetation. Then, the detection algorithm identified this person as a new object and assigned him to a new tracker and new id tracking number. In row of EXP5 of Table 3 the same scenario was reproduced multiple times, significantly increasing the IDC metric. It should be noted that, by default, the id tracking number of the first detected person in the video feed was randomly selected as an integer number. Then, the id tracking number is increased by one (1) for the next detected and tracked person and so on. Such setting can be easily modified and parameterized according to use case and possible multiple objects.

4. Conclusions

According to recent trends in smart cities, situational awareness is considered as a critical aspect of the decision-making process in emergency response, public safety, and civil protection, and it requires the availability of up-to-date information on the current situation. To this end, this study presents a Video Analytics Edge Computing (VAEC) system that performs real-time enhanced situation awareness for person detection in a video surveillance manner. Once a person (or persons) is detected, the VAEC system is able to capture relevant crucial information, such as timestamp, geolocation, number of the detected persons, and video frames with the detections. Such information is able to be aggregated and shared via the proposed Distribute Edge Computing Internet of Things (DECIoT) platform (that is integrated to VAEC's edge device) to other smart city platforms via proper middleware. The DECIoT is based on the EdgeX foundry open source framework, which is considered as a highly flexible and scalable edge computing framework.

The VAEC's embedded edge device adopts GPU processing and executes state-of-the-art object detection and tracking-by-detection (TBD) algorithms (i.e., pre-trained YOLOv5 and DeepSort, respectively). To verify the utility and functionality of the VAEC system, extended experiments were performed. More specifically, three RGB camera sensors and one thermal camera sensor were utilized to investigate the performance of the pre-trained YOLOv5 (only the class "person" of the pre-trained YOLOv5 was activated) and DeepSort, in real-world scenarios, in different light conditions (day and night). Several viewing perspectives of two uses cases were also considered, i.e., installing the VAEC's edge device in a fixed position (to a building of 10 m height, with the camera looking in outdoor view) and mounting to a car. For the building case, the maximum distance between the camera and the object associated with an effective and stable person detection was 45 m. For the moving car case, the maximum distance between the camera and the object associated with an effective and stable person detection was 45 m for the CAM1 (RGB) and CAM4 (thermal), and it was 35 m for the CAM3 (RGB). Although CAM3 has a wide view of the scene (due to its wide angle len), the objects are depicted smaller in the video frames compared to the corresponding ones of CAM1 and CAM4, decreasing its performance at larger distances. Objective criteria were adopted to quantitatively evaluate the object detection and TBD processes. The achieved results of the YOLOv5s, for all the cases and camera sensors (average values of $C_M = 82.8\%$, $C_R = 91.7\%$, $Q = 76.4\%$, and $F1 = 86.4\%$), are considered to be satisfactory, proving its suitability and efficiency for such applications with several challenges. The observed False Positive (FP) entries were mainly due to artificial patterns, or objects with similar patterns with persons, which led to the reduction in the C_R rate. On the other hand, the observed FN entries were mainly due to the large distance

between the camera and person (resulting in a small sized object that is unrecognizable by the algorithm) or with similar spectral values between background and the person.

Concerning the TBD process, the achieved results of the DeepSort are considered to be satisfactory, proving its stability and effectiveness for tracking objects under a TBD manner. However, for cases with occlusions, the tracker lost the tracked person, leading to the change of the corresponding id tracking number after the associated detection, increasing the IDC metric. To fix such issues, a re-identification process (through which lost objects are able to be re-detected and re-tracked), and a fusion of spatio-temporal and visual information to improve re-identification accuracy, is required. Such re-identification algorithms and fusion analyses will be considered in the VAEC system in future work. Although, in this study, the VAEC is used for a use case of abnormal detection during surveillance, i.e., detection of a person in a prohibited area, the system is potentially extendable by activating multiple classes of other objects of interests (e.g., vehicles, etc.), as well as fully modular to be adapted to other applications, e.g., search and rescue, etc. The performance of custom weights of the YOLOv5 deep learning model will also be investigated in future work, carrying out a new training process and adjusting the model's parameters only for persons (or for only a subset of expected objects that need to be detected in the scenario). As future work, we intend to perform further extended experimental analysis for the pre-trained YOLOv5 and a custom YOLOv5, in terms of real-world applications, with poor weather conditions and more complex scenes with variable perspectives of the objects [76–78].

Finally, representative experimental results were presented that verify the effective integration of the DECIoT to the edge device, sharing information of the person detection alert to Apache Kafka. In this study, the Apache Kafka was considered the middleware of a smart city platform. Hence, representative experimental results were presented, indicating the route that the information passes through in each layer and microservice of DECIoT, i.e., from southbound physical devices to northbound external middleware. The results highlighted the potential of the VAEC system to be exploited by decision-makers or city authorities, providing enhanced situational awareness.

Author Contributions: Conceptualization, E.M., P.L., A.D., L.K., E.O. and M.K.; methodology, E.M., P.L., A.D. and L.K.; software, P.L. and A.D.; validation, E.M., P.L., A.D., L.K. and M.K.; writing—review and editing, E.M., P.L., A.D., L.K., E.O. and M.K.; supervision, E.M., L.K., E.O., M.K. and A.A. All authors have read and agreed to the published version of the manuscript.

Funding: This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 883522.

Institutional Review Board Statement: No ethical approval required for this study. This research complies with H2020 Regulation (No 1291/2013 EU), particularly with Article 19 “Ethical Principles” in the context of the Smart Spaces Safety and Security for All Cities—S4AllCities project (under grant agreement No 883522).

Informed Consent Statement: Informed consent was obtained from all subjects involved in the study.

Data Availability Statement: Data sharing is not applicable to this article.

Acknowledgments: This work is a part of the S4AllCities project. This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 883522. Content reflects only the authors' view and the Research Executive Agency (REA)/European Commission is not responsible for any use that may be made of the information it contains.

Conflicts of Interest: The authors declare no conflict of interest.

References

- Smagilova, E.; Hughes, L.; Rana, N.P.; Dwivedi, Y.K. Security, Privacy and Risks Within Smart Cities: Literature Review and Development of a Smart City Interaction Framework. *Inf. Syst. Front.* **2020**, 1–22. [\[CrossRef\]](#)
- Voas, J. Demystifying the Internet of Things. *Computer* **2016**, 49, 80–83. [\[CrossRef\]](#)
- SINTEF. New Waves of IoT Technologies Research—Transcending Intelligence and Senses at the Edge to Create Multi Experience Environments. Available online: <https://www.sintef.no/publikasjoner/publikasjon/1896632/> (accessed on 11 January 2022).
- Zanella, A.; Bui, N.; Castellani, A.; Vangelista, L.; Zorzi, M. Internet of Things for Smart Cities. *IEEE Internet Things J.* **2014**, 1, 22–32. [\[CrossRef\]](#)
- Endsley, M.R. Toward a Theory of Situation Awareness in Dynamic Systems. *Hum. Factors J. Hum. Factors Ergon. Soc.* **1995**, 37, 32–64. [\[CrossRef\]](#)
- Rummukainen, L.; Oksama, L.; Timonen, J.; Vankka, J.; Lauri, R. Situation awareness requirements for a critical infrastructure monitoring operator. In Proceedings of the 2015 IEEE International Symposium on Technologies for Homeland Security (HST), Greater Boston, MA, USA, 14–16 April 2015; pp. 1–6. [\[CrossRef\]](#)
- Geraldes, R.; Goncalves, A.; Lai, T.; Villerabel, M.; Deng, W.; Salta, A.; Nakayama, K.; Matsuo, Y.; Prendinger, H. UAV-Based Situational Awareness System Using Deep Learning. *IEEE Access* **2019**, 7, 122583–122594. [\[CrossRef\]](#)
- Eräranta, S.; Staffans, A. From Situation Awareness to Smart City Planning and Decision Making. In Proceedings of the 14th International Conference on Computers in Urban Planning and Urban Management, Cambridge, MA, USA, 7–10 July 2015.
- Suri, N.; Zielinski, Z.; Tortonesi, M.; Fuchs, C.; Pradhan, M.; Wrona, K.; Furtak, J.; Vasilache, D.B.; Street, M.; Pellegrini, V.; et al. Exploiting Smart City IoT for Disaster Recovery Operations. In Proceedings of the 2018 IEEE 4th World Forum on Internet of Things (WF-IoT), Singapore, 5–8 February 2018; pp. 458–463. [\[CrossRef\]](#)
- Mitaritonna, A.; Abasolo, M.J.; Montero, F. Situational Awareness through Augmented Reality: 3D-SA Model to Relate Requirements, Design and Evaluation. In Proceedings of the 2019 International Conference on Virtual Reality and Visualization (ICVRV), Hong Kong, China, 18–19 November 2019; pp. 227–232. [\[CrossRef\]](#)
- Neshenko, N.; Nader, C.; Bou-Harb, E.; Furht, B. A survey of methods supporting cyber situational awareness in the context of smart cities. *J. Big Data* **2020**, 7, 92. [\[CrossRef\]](#)
- Jiang, M. An Integrated Situational Awareness Platform for Disaster Planning and Emergency Response. In Proceedings of the 2020 IEEE International Smart Cities Conference (ISC2), Online, 28 September–1 October 2020; pp. 1–6. [\[CrossRef\]](#)
- Sun, H.; Shi, W.; Liang, X.; Yu, Y. VU: Edge Computing-Enabled Video Usefulness Detection and its Application in Large-Scale Video Surveillance Systems. *IEEE Internet Things J.* **2019**, 7, 800–817. [\[CrossRef\]](#)
- Github. Ultralytics. Ultralytics/Yolov5. 2022. Available online: <https://github.com/ultralytics/yolov5> (accessed on 13 January 2022).
- Griffin, B.A.; Corso, J.J. BubbleNets: Learning to Select the Guidance Frame in Video Object Segmentation by Deep Sorting Frames. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 8906–8915. [\[CrossRef\]](#)
- Patrikar, D.R.; Parate, M.R. Anomaly Detection using Edge Computing in Video Surveillance System: Review. *arXiv* **2021**, arXiv:2107.02778. Available online: <http://arxiv.org/abs/2107.02778> (accessed on 13 January 2022).
- Liu, Z.; Chen, Z.; Li, Z.; Hu, W. An Efficient Pedestrian Detection Method Based on YOLOv2. *Math. Probl. Eng.* **2018**, 2018, 3518959. [\[CrossRef\]](#)
- Ahmad, M.; Ahmed, I.; Adnan, A. Overhead View Person Detection Using YOLO. In Proceedings of the 2019 IEEE 10th Annual Ubiquitous Computing, Electronics Mobile Communication Conference (UEMCON), New York, NY, USA, 10–12 October 2019; pp. 0627–0633. [\[CrossRef\]](#)
- Karthikeyan, B.; Lakshmanan, R.; Kabilan, M.; Madeshwaran, R. Real-Time Detection and Tracking of Human Based on Image Processing with Laser Pointing. In Proceedings of the 2020 International Conference on System, Computation, Automation and Networking (ICSCAN), Pondicherry, India, 3–4 July 2020; pp. 1–5. [\[CrossRef\]](#)
- Matveev, I.; Karpov, K.; Chmielewski, I.; Siemens, E.; Yurchenko, A. Fast Object Detection Using Dimensional Based Features for Public Street Environments. *Smart Cities* **2020**, 3, 93–111. [\[CrossRef\]](#)
- Kim, C.; Oghaz, M.; Fajtl, J.; Argyriou, V.; Remagnino, P. A Comparison of Embedded Deep Learning Methods for Person Detection. In Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, Prague, Czech Republic, 25–27 February 2019; Volume 5, pp. 459–465. [\[CrossRef\]](#)
- Farooq, M.A.; Corcoran, P.; Rotariu, C.; Shariff, W. Object Detection in Thermal Spectrum for Advanced Driver-Assistance Systems (ADAS). *IEEE Access* **2021**, 9, 156465–156481. [\[CrossRef\]](#)
- Chen, J.; Wang, F.; Li, C.; Zhang, Y.; Ai, Y.; Zhang, W. Online Multiple Object Tracking Using a Novel Discriminative Module for Autonomous Driving. *Electronics* **2021**, 10, 2479. [\[CrossRef\]](#)
- Wu, H.; Du, C.; Ji, Z.; Gao, M.; He, Z. SORT-YM: An Algorithm of Multi-Object Tracking with YOLOv4-Tiny and Motion Prediction. *Electronics* **2021**, 10, 2319. [\[CrossRef\]](#)
- Rajavel, R.; Ravichandran, S.K.; Harimoorthy, K.; Nagappan, P.; Gobichettipalayam, K.R. IoT-based smart healthcare video surveillance system using edge computing. *J. Ambient Intell. Humaniz. Comput.* **2021**. [\[CrossRef\]](#)
- Xu, R.; Nikouei, S.Y.; Chen, Y.; Polunchenko, A.; Song, S.; Deng, C.; Faughnan, T.R. Real-Time Human Objects Tracking for Smart Surveillance at the Edge. In Proceedings of the 2018 IEEE International Conference on Communications (ICC), Kansas City, MO, USA, 20–24 May 2018; pp. 1–6. [\[CrossRef\]](#)

27. Maltezos, E.; Protopapadakis, E.; Doulamis, N.; Doulamis, A.; Ioannidis, C. Understanding Historical Cityscapes from Aerial Imagery through Machine Learning. In *Digital Heritage. Progress in Cultural Heritage: Documentation, Preservation, and Protection, Proceedings of the Euro-Mediterranean Conference, Nicosia, Cyprus, 29 October–3 November 2018*; Springer: Cham, Switzerland, 2018; pp. 200–211. [\[CrossRef\]](#)
28. Maltezos, E.; Doulamis, A.; Ioannidis, C. Improving the Visualisation of 3D Textured Models via Shadow Detection and Removal. In *Proceedings of the 2017 9th International Conference on Virtual Worlds and Games for Serious Applications (VS-Games), Athens, Greece, 6–8 September 2017*; pp. 161–164. [\[CrossRef\]](#)
29. Pudasaini, D.; Abhari, A. Scalable Object Detection, Tracking and Pattern Recognition Model Using Edge Computing. In *Proceedings of the 2020 Spring Simulation Conference (SpringSim), Fairfax, VA, USA, 19–21 May 2020*; pp. 1–11. [\[CrossRef\]](#)
30. Pudasaini, D.; Abhari, A. Edge-based Video Analytic for Smart Cities. *Int. J. Adv. Comput. Sci. Appl.* **2021**, *12*, 10. [\[CrossRef\]](#)
31. Kafka. Apache Kafka. Available online: <https://kafka.apache.org/> (accessed on 26 January 2022).
32. Nasiri, H.; Nasehi, S.; Goudarzi, M. A Survey of Distributed Stream Processing Systems for Smart City Data Analytics. In *Proceedings of the International Conference on Smart Cities and Internet of Things—SCIOT '18, Mashhad, Iran, 26–27 September 2018*; pp. 1–7. [\[CrossRef\]](#)
33. Chen, A.T.-Y.; Biglari-Abhari, M.; Wang, K.I.-K. Trusting the Big Computer in Computer Vision: A Privacy-Affirming Framework. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Honolulu, HI, USA, 21–26 July 2017*; pp. 1360–1367. [\[CrossRef\]](#)
34. Zhao, Y.; Yin, Y.; Gui, G. Lightweight Deep Learning Based Intelligent Edge Surveillance Techniques. *IEEE Trans. Cogn. Commun. Netw.* **2020**, *6*, 1146–1154. [\[CrossRef\]](#)
35. Bochkovskiy, A.; Wang, C.; Liao, H. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv* **2004**, arXiv:abs/2004.10934. Available online: <http://arxiv.org/abs/2004.10934> (accessed on 13 January 2022).
36. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016*; pp. 779–788. [\[CrossRef\]](#)
37. Maltezos, E.; Douklias, A.; Dadoukis, A.; Misichroni, F.; Karagiannidis, L.; Antonopoulos, M.; Voulgary, K.; Ouzounoglou, E.; Amditis, A. The INUS Platform: A Modular Solution for Object Detection and Tracking from UAVs and Terrestrial Surveillance Assets. *Computation* **2021**, *9*, 12. [\[CrossRef\]](#)
38. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. In *Computer Vision—ECCV 2016, Proceedings of the 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016*; Leibe, B., Matas, J., Sebe, N., Welling, M., Eds.; Springer International Publishing: Cham, Switzerland, 2016; pp. 21–37. [\[CrossRef\]](#)
39. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014*; pp. 580–587. [\[CrossRef\]](#)
40. Girshick, R. Fast R-CNN. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015*; pp. 1440–1448. [\[CrossRef\]](#)
41. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *42*, 386–397. [\[CrossRef\]](#)
42. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft COCO: Common Objects in Context. In *Proceedings of the 13th European Conference, Zurich, Switzerland, 6–12 September 2014*; pp. 740–755.
43. Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; Fei-Fei, L. ImageNet: A Large-Scale Hierarchical Image Database. In *Proceedings of the 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009*; pp. 248–255. [\[CrossRef\]](#)
44. Everingham, M.; Van Gool, L.; Williams, C.K.I.; Winn, J.; Zisserman, A. The Pascal Visual Object Classes (VOC) Challenge. *Int. J. Comput. Vis.* **2010**, *88*, 303–338. [\[CrossRef\]](#)
45. Kalal, Z.; Mikolajczyk, K.; Matas, J. Forward-Backward Error: Automatic Detection of Tracking Failures. In *Proceedings of the 2010 20th International Conference on Pattern Recognition, Istanbul, Turkey, 23–26 August 2010*; pp. 2756–2759. [\[CrossRef\]](#)
46. Grabner, H.; Grabner, M.; Bischof, H. Real-Time Tracking via On-line Boosting. In *Proceedings of the British Machine Vision Conference, Edinburgh, UK, 4–7 September 2006*.
47. Held, D.; Thrun, S.; Savarese, S. Learn to Track at 100 fps with Deep Regression Networks. In *Computer Vision—ECCV 2016*; Springer: Cham, Switzerland, 2016; Volume 9905, pp. 749–765. [\[CrossRef\]](#)
48. Bolme, D.S.; Beveridge, J.R.; Draper, B.A.; Lui, Y.M. Visual Object Tracking Using Adaptive Correlation Filters. In *Proceedings of the 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Francisco, CA, USA, 13–18 June 2010*; pp. 2544–2550. [\[CrossRef\]](#)
49. Lukežič, A.; Vojř, T.; Zajc, L.; Matas, J.; Kristan, M. Discriminative Correlation Filter Tracker with Channel and Spatial Reliability. *Int. J. Comput. Vis.* **2018**, *126*, 671–688. [\[CrossRef\]](#)
50. Kalal, Z.; Mikolajczyk, K.; Matas, J. Face-TLD: Tracking-Learning-Detection Applied to Faces. In *Proceedings of the 2010 IEEE International Conference on Image Processing, Hong Kong, China, 27–29 September 2010*; pp. 3789–3792. [\[CrossRef\]](#)
51. Cai, C.; Liang, X.; Wang, B.; Cui, Y.; Yan, Y. A Target Tracking Method Based on KCF for Omnidirectional Vision. In *Proceedings of the 2018 37th Chinese Control Conference (CCC), Wuhan, China, 25–27 July 2018*; pp. 2674–2679. [\[CrossRef\]](#)

52. Babenko, B.; Yang, M.-H.; Belongie, S. Visual Tracking with Online Multiple Instance Learning. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 983–990. [\[CrossRef\]](#)
53. Zhang, Y.; Wang, C.; Wang, X.; Zeng, W.; Liu, W. FairMOT: On the Fairness of Detection and Re-identification in Multiple Object Tracking. *Int. J. Comput. Vis.* **2021**, *129*, 3069–3087. [\[CrossRef\]](#)
54. Zhao, Z.-Q.; Zheng, P.; Xu, S.-T.; Wu, X. Object Detection With Deep Learning: A Review. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, *30*, 3212–3232. [\[CrossRef\]](#)
55. Zhu, X.; Lyu, S.; Wang, X.; Zhao, Q. TPH-YOLOv5: Improved YOLOv5 Based on Transformer Prediction Head for Object Detection on Drone-captured Scenarios. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW), Montreal, QC, Canada, 11–17 October 2021; pp. 2778–2788. [\[CrossRef\]](#)
56. Nepal, U.; Eslamiat, H. Comparing YOLOv3, YOLOv4 and YOLOv5 for Autonomous Landing Spot Detection in Faulty UAVs. *Sensors* **2022**, *22*, 464. [\[CrossRef\]](#)
57. Phadtare, M.; Choudhari, V.; Pedram, R.; Vartak, S. Comparison between YOLO and SSD Mobile Net for Object Detection in a Surveillance Drone. *Int. J. Sci. Res. Eng. Man.* **2021**, *5*. [\[CrossRef\]](#)
58. Zhou, F.; Zhao, H.; Nie, Z. Safety Helmet Detection Based on YOLOv5. In Proceedings of the 2021 IEEE International Conference on Power Electronics, Computer Applications (ICPECA), Shenyang, China, 22–24 January 2021; pp. 6–11. [\[CrossRef\]](#)
59. Kaya, A.; Keceli, A.S.; Catal, C.; Yalic, H.Y.; Temucin, H.; Tekinerdogan, B. Analysis of transfer learning for deep neural network based plant classification models. *Comput. Electron. Agric.* **2019**, *158*, 20–29. [\[CrossRef\]](#)
60. Mike, *Yolov5 + Deep Sort with PyTorch*. 2022. Available online: https://github.com/mikel-brostrom/Yolov5_DeepSort_Pytorch (accessed on 1 February 2022).
61. TensorFlow. Libraries & Extensions. Available online: <https://www.tensorflow.org/resources/libraries-extensions> (accessed on 13 January 2022).
62. “Home” OpenCV. Available online: <https://opencv.org/> (accessed on 13 January 2022).
63. T. L. Foundation. Welcome. Available online: <https://www.edgexfoundry.org> (accessed on 13 January 2022).
64. Maltezos, E.; Karagiannidis, L.; Dadoukis, A.; Petousakis, K.; Misichroni, F.; Ouzounoglou, E.; Gounaridis, L.; Gounaridis, D.; Kouloumentas, C.; Amditis, A. Public Safety in Smart Cities under the Edge Computing Concept. In Proceedings of the 2021 IEEE International Mediterranean Conference on Communications and Networking (MeditCom), Athens, Greece, 7–10 September 2021; pp. 88–93. [\[CrossRef\]](#)
65. Suci, G.; Hussain, I.; Iordache, G.; Beceanu, C.; Kecs, R.A.; Vochin, M.-C. Safety and Security of Citizens in Smart Cities. In Proceedings of the 2021 20th RoEduNet Conference: Networking in Education and Research (RoEduNet), Iasi, Romania, 4–6 November 2021; pp. 1–8. [\[CrossRef\]](#)
66. Cao, K.; Liu, Y.; Meng, G.; Sun, Q. An Overview on Edge Computing Research. *IEEE Access* **2020**, *8*, 85714–85728. [\[CrossRef\]](#)
67. Jin, S.; Sun, B.; Zhou, Y.; Han, H.; Li, Q.; Xu, C.; Jin, X. Video Sensor Security System in IoT Based on Edge Computing. In Proceedings of the 2020 International Conference on Wireless Communications and Signal Processing (WCSP), Wuhan, China, 20–23 October 2020; pp. 176–181. [\[CrossRef\]](#)
68. Villali, V.; Bijivemula, S.; Narayanan, S.L.; Prathusha, T.M.V.; Sri, M.S.K.; Khan, A. Open-Source Solutions for Edge Computing. In Proceedings of the 2021 2nd International Conference on Smart Electronics and Communication (ICOSEC), Tiruchirappalli, India, 7–9 October 2021; pp. 1185–1193. [\[CrossRef\]](#)
69. Xu, R.; Jin, W.; Kim, D. Enhanced Service Framework Based on Microservice Management and Client Support Provider for Efficient User Experiment in Edge Computing Environment. *IEEE Access* **2021**, *9*, 110683–110694. [\[CrossRef\]](#)
70. The Alliance for the Internet of Things Innovation. Available online: <https://aioti.eu/> (accessed on 13 January 2022).
71. The Go Programming Language. Available online: <https://go.dev/> (accessed on 1 February 2022).
72. Eclipse Mosquitto. *Eclipse Mosquitto*, 8 January 2018. Available online: <https://mosquitto.org/> (accessed on 26 January 2022).
73. Rottensteiner, F.; Sohn, G.; Gerke, M.; Wegner, J.D. ISPRS Test Project on Urban Classification and 3D Building Reconstruction. 2013. Available online: http://www2.isprs.org/tl_files/isprs/wg34/docs/ComplexScenes_revision_v4.pdf (accessed on 1 February 2022).
74. Wen, C.; Li, X.; Yao, X.; Peng, L.; Chi, T. Airborne LiDAR point cloud classification with global-local graph attention convolution neural network. *ISPRS J. Photogramm. Remote Sens.* **2021**, *173*, 181–194. [\[CrossRef\]](#)
75. Godil, A.; Shackleford, W.; Shneier, M.; Bostelman, R.; Hong, T. *Performance Metrics for Evaluating Object and Human Detection and Tracking Systems*; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2014. [\[CrossRef\]](#)
76. Sharma, T.; Debaque, B.; Duclos, N.; Chehri, A.; Kinder, B.; Fortier, P. Deep Learning-Based Object Detection and Scene Perception under Bad Weather Conditions. *Electronics* **2022**, *11*, 563. [\[CrossRef\]](#)
77. Montenegro, B.; Flores, M. Pedestrian Detection at Daytime and Nighttime Conditions Based on YOLO-v5—Ingenius. 2022. Available online: <https://ingenius.ups.edu.ec/index.php/ingenius/article/view/27.2022.08> (accessed on 1 February 2022).
78. Li, S.; Li, Y.; Li, Y.; Li, M.; Xu, X. YOLO-FIRI: Improved YOLOv5 for Infrared Image Object Detection. *IEEE Access* **2021**, *9*, 141861–141875. [\[CrossRef\]](#)