future internet

MDPI

Article

An Advanced Deep Learning Approach for Multi-Object Counting in Urban Vehicular Environments

Ahmed Dirir [1], Henry Ignatious [1], Hesham Elsayed [1,2,*], Manzoor Khan [1,2], Mohammed Adib [1], Anas Mahmoud [1] and Moatasem Al-Gunaid [1]

[1] College of Information Technology, UAE University, Al Ain, Abu Dhabi 15551, United Arab Emirates; 100057669@ku.ac.ae (A.D.); halex@uaeu.ac.ae (H.I.); manzoor-khan@uaeu.ac.ae (M.K.); 201640065@uaeu.ac.ae (M.A.); 201540050@uaeu.ac.ae (A.M.); 201450042@uaeu.ac.ae (M.A.-G.)
[2] Emirates Center for Mobility Research, UAE University, Al Ain, Abu Dhabi 15551, United Arab Emirates
* Correspondence: helsayed@uaeu.ac.ae

Abstract: Object counting is an active research area that gained more attention in the past few years. In smart cities, vehicle counting plays a crucial role in urban planning and management of the Intelligent Transportation Systems (ITS). Several approaches have been proposed in the literature to address this problem. However, the resulting detection accuracy is still not adequate. This paper proposes an efficient approach that uses deep learning concepts and correlation filters for multi-object counting and tracking. The performance of the proposed system is evaluated using a dataset consisting of 16 videos with different features to examine the impact of object density, image quality, angle of view, and speed of motion towards system accuracy. Performance evaluation exhibits promising results in normal traffic scenarios and adverse weather conditions. Moreover, the proposed approach outperforms the performance of two recent approaches from the literature.

Keywords: object counting; object detection; multi-object tracking; deep learning; YOLO; correlation filters

Citation: Dirir, A.; Ignatious, H.; Elsayed, H.; Khan, M.; Adib, M.; Mahmoud, A.; Al-Gunaid, M. An Efficient Multi-Object Tracking and Counting System Using Deep Learning in Urban Vehicular Environments. *Future Internet* **2021**, *13*, 306. https://doi.org/10.3390/fi13120306

Academic Editors: Khalid Elgazzar and Aboelmagd Noureldin

Received: 3 November 2021
Accepted: 28 November 2021
Published: 30 November 2021

1. Introduction

Drones are widely used in many emerging fields such as military, disaster monitoring and recovery, outer space, transportation, wildlife and historical conservation, medicine, agriculture and photography [1–4]. Drones must be equipped with smart computer vision and autopilot in order to be widely deployed and to further lower costs. Object detection and tracking are critical for catching crucial elements in a picture while using aerial photography. In a variety of fields, such as traffic data gathering, traffic monitoring, film and television shooting, visual object tracking in an unmanned aerial vehicle (UAV) recording plays a significant role. However, various issues such as appearance variation, background clutter, and extreme occlusion make it difficult to track the target reliably in a UAV vision task. In recent years, UAV-based low-altitude aerial photography technology has been widely deployed as a viable supplement to aircraft remote sensing and satellite remote sensing for traffic data collection. This device uses aerial high-resolution cameras to clearly capture ground objects, with imaging resolution reaching centimeters. As a result, the volume of traffic image data provided by unmanned aerial vehicles' low-altitude aerial photography has skyrocketed. Manually processing a big amount of data is not only time-consuming but also inefficient. As a result, intelligent processing of UAV pictures has become a hotspot for research. As one of the technologies, single object tracking of UAV photos offers the foundation for later tasks such as vehicle traffic assessment and road conflict prediction. In single object tracking, the target's initial position is manually established in the first frame, and the target's bounding box is continually predicted in the following frames. Although significant progress has been made in recent years, effec-

tive tracking remains a difficult challenge in UAV circumstances involving appearance fluctuation, background clutter, and extreme occlusion

Many applications require the knowledge of object counts in the environment to function properly or plan beforehand. For instance, transportation agencies can optimize their infrastructure plans if they have an accurate measure of traffic flow on each road [5,6]. Counting methods can be used in traffic monitoring to track the number of moving autos, pedestrians, and parked cars. Of course, they cannot install sophisticated equipment on each road to achieve such goals. Instead, they are looking for cheap alternatives that can give accurate results. Security, surveillance, access point management, urban planning, traffic control, business, and industrial applications all require automatic visual recognition, tracking, and counting of a variable number of objects. These applications, however, failed to play a significant role in consumer electronics. The main reason is that they demand stringent criteria in order to create good working conditions, specialized and costly hardware, complex installation and setup procedures, and qualified worker supervision.

Road users detection, tracking, and counting are very crucial for enabling autonomous behavior in the envisioned intelligent transportation systems [7]. Accurate object detection is important for other application domains as well. For instance, detecting and counting the number of olive fruit flies. It is important to early detect and count the presence of olive fruit flies as it can damage up to 100% of the harvested fruit and can cause up to 80% reduction of the value of the resulting olive oil [8]. They can also be used to keep track of the number of different species, such as penguins, which is crucial for wildlife conservation.

Counting objects presents many difficulties. The variety of the items in terms of shape, size, attitude, and appearance must be learned by the models. Objects may also be partially obscured and appear at varying angles and resolutions. Furthermore, the background, weather conditions, and illuminations might differ significantly between scenarios. In order to perform effective object counting, the model must be strong enough to distinguish objects in the presence of these fluctuations.

In our previous paper [9], we proposed an object counting system that used the YOLO model for object detection and the Kernelised Correlation Filter (KCF) for object tracking. The system has exhibited a fast processing time and reasonable accuracy. In this paper, we extend our previous work and propose a more efficient system for object tracking and counting which integrates the latest version of YOLO (Yolo5) for object detection and the Channel and Spatial Reliability Tracker (CSRT) for object tracking and counting. The latest versions of YOLO improve the accuracy of object detection and reduce the required processing time. Moreover, the CSRT tracker improves accuracy over the KCF tracker, which in turn reduces the counting errors.

The content of this paper is organized as follows. Section 2 provides the background on YOLO and CSRT. Section 3 discusses the related work. Section 4 describes the function and implementation of the proposed architecture. Section 5 presents the strategies used for preparing and analyzing the datasets. Section 6 evaluates and analyzes the performance of the proposed model. Section 7 discusses the future work, and Section 8 concludes the paper.

2. Background

Object detection algorithms can be divided into two main categories, algorithms based on classification and algorithms based on regression. Classification-based algorithms are executed in two phases, the first phase where they select the regions of interest, and the second phase where they perform classification of these regions using convolutional neural networks. This approach is slow because we have to classify each region. A widely known example of this type of algorithm is the RCNN. Regression-based algorithms assign the bounding boxes and perform the predictions of classes in one shot. As mentioned earlier, the two best-known algorithms from this group are the SSD and YOLO. Regression-based algorithms are best suited for real-time applications where we can compromise the accuracy a little bit to gain faster processing speed.

YOLO deep neural network consists of several convolutional layers followed by fully connected layers. Convolutional layers work as feature extractors that will extract the required features from the image. There is another version of YOLO called Fast YOLO. It has fewer convolutional layers and fewer filters in these convolutional layers. Since the number of layers is fewer, Fast YOLO is faster than YOLO, however, YOLO has better accuracy. The YOLO algorithm tries to predict a class of an object and the bounding box specifying the location of the object. The YOLO algorithm divides the image into cells of predefined sizes where each cell can predict up to five objects. Since there might be more than one object in the same cell, YOLO allows each cell to detect up to five objects. For instance, if there are a table, a cup, and a cat at the same location, we will get one of the three boxes, either for the car, the cup, or for the cat. To detect all objects, anchor boxes are used to specify bounding boxes with different aspect ratios. The aspect ratios are calculated by applying the k-means clustering algorithm on all bounding boxes in the training dataset.

The number of detected objects in an image is limited because of YOLO architecture. For instance, if we have a 13×13 grid of cells where each cell detects five different objects, we can predict up to 845 bounding boxes for one image. Typically, most of these cells will be empty and will not contain any objects. Thus, YOLO assigns a value pc for each cell that will be used to filter out cells that do not contain any objects. Another issue that might arise in such settings is the overlap between detected objects, YOLO can detect five bounding boxes per cell, and these cells are adjacent to each other, there might be multiple bounding boxes on the same object. To solve this issue, YOLO uses non-max suppression to choose the bounding box with the highest pc value. For each cell, YOLO checks if there is an object, then, searches for all bounding boxes that refer to the same object and selects the object with the highest pc value. The non-max suppression is extremely important for our application. As we are counting objects and relying on YOLO to detect these objects. The number of predicted bounding boxes affects the total counts, thus, it is mandatory to ensure redundant bounding boxes are removed.

Since there is a spatial constraint imposed by YOLO whereby each cell is detecting five objects, YOLO algorithms performance degrades with small objects that are close to each other. The most important output for our application is the bounding boxes that YOLO detects. These boxes will be fed from the object detection stage to the object tracking stage. YOLO uses descriptors for each bounding box, the main descriptors are:

- Center of a bounding box.
- Width and Height.
- The value (c) which is corresponding to a class of an object.
- The (pc) value, which is the probability that there is an object in the bounding box.

Correlation filters are broadly applied in several applications. Correlations filters examine different samples to find the correlation between them. Correlation refers to similarity in general terms. In the case of object tracking, the samples are the images or a particular segment of an image. If two images are similar, the correlation between them is high, if they are not similar to each other, the correlation between the two images is small.

The CSRT tracker is a C++ implementation of the CSR-DCF (Channel and Spatial Reliability of Discriminative Correlation Filter) tracking algorithm in the OpenCV library [10]. Trackers are utilities, which are used to detect, identify and track the actual objects embedded within a frame. Object detection is performed once, while the object tracker handles the rest of the frames, resulting in a quicker and more efficient object-tracking pipeline. The spatial reliability map is used in the Discriminative Correlation Filter with Channel and Spatial Reliability (DCF-CSR) to modify the filter support which is the part of the selected area from the frame for tracking. This results in the specified region being enlarged and localized, as well as enhanced tracking of non-rectangular regions for objects. It merely makes use of two common features (Histogram of oriented grading (HoGs) and Color names). It also has a reduced frame rate (25 fps) but provides better object tracking precision. Open CV API uses eight types of filters for effective feature extraction and object

detection. The most used filters are (i) BOOSING Tracker (ii) MIL Tracker (iii) KCF Tracker (iv) CSRT Tracker (v) Median Flow Tracker (vi) TLD Tracker (vii) MOSSE Tracker (viii) GOTURn Tracker. Among the listed object trackers, this study uses a CSRT tracker, due to its extended ability in identifying the features of the detected objects within a frame and the high rate of accuracy in identifying and tracking the objects.

3. Related Work

This section analyses various articles proposed by eminent authors related to object detection and counting in multiple domains, identifies their significant contributions, and pinpoints the pitfalls associated with the existing literature.

3.1. Object Detection and Counting Using Traditional Approaches

Object detection, tracking, and counting have been active research areas that gained more attention in the past few years. Some research has concentrated on building automatic detection and tracking systems that reduce the need for supervision. They frequently use background subtraction-based [11] moving object detectors because these detectors do not require a training stage or complicated system parameter setup. Several researchers have proposed and used versatile models to track, detect and count objects for various reasons. The authors have used models that use advanced probability concepts and ML approaches to detect, track, and count the objects effectively. However, these detectors have several flaws that result in false tracking, noisy and missing detections, and split and merged detections [12]. Furthermore, the relationship between detected and tracked objects is uncertain. Various data association approaches have been proposed to tackle these challenges. To represent possible split and merged occurrences, a collection of detected objects is augmented by virtual detections [13]. Some literature employs an overlapping criterion to ease the development of virtual detections [14].

In [15], the authors proposed a probabilistic model for simulating split and merged detections, which uses a batch procedure to compute association hypotheses using the Markov Chain Monte Carlo method (MCMC). For a fixed number of objects, ref. [16] proposes a sequential technique based on MCMC sampling, while ref. [17] describes a similar strategy for a variable number of objects. The fundamental drawback of earlier methods is that they have a high computational cost and require specialized infrastructure to perform object detections in real-time. Other approaches [18,19] restrict data association and tracking difficulties by using prior information about the geometry of the scene, such as the floor location and camera calibration. However, because the camera must be calibrated and the 3D plane of the floor must be estimated based on the camera position, this approach makes the system installation and setup difficult.

For interacting targets, the authors [20] offer a multiple hypothesis tracker that generates split and merged data. The tracker is built on an efficient auxiliary variable particle filter based on Markov chain Monte Carlo (MCMC). Further investigations are needed to monitor the models' behavior to further improve the tracking efficiency and data association occurring during object interaction. The authors proposed a Bayesian Visual Surveillance Model that can handle unwanted measurements. Split and merged measurements, in particular, are explicitly characterized by stochastic processes. A particle filtering technique that blends traditional and MCMC sampling allows for reliable inference. The major problem associated with the authors' approach is the complexity associated with the Bayesian approach, due to which additional filters are used to combine traditional and MCMC sampling techniques, which improves the accuracy of simulating data association between Multiscale Residual (MR) and tracked objects.

In [21], proposes a unique object detection and tracking model based on learning Existing techniques which operate by linearizing the motion that makes an implicit assumption about Euclidean space. The majority of computer vision transformations have a matrix Lie group structure. The learning model is expanded to train a class-specific tracking function, which is then combined with a pose-dependent object detector to create a pose-invariant

object identification system. When compared to the existing object identification approach, the suggested model can reliably recognize objects in various postures, while the search space is just a fraction of the size. The drawback associated with the authors' proposed approach is the cost associated with the experimental setup and more time consumption to initially train the model.

3.2. Object Detection and Counting Using Deep Learning Techniques

Object detection in photos, which seeks to recognize things from a specified set of object categories (e.g., cars and pedestrians), is a long-standing topic. Many applications, such as image interpretation, video monitoring, and anomaly detection, would benefit from accurate object detection right away. Counting objects is a relatively common task in a variety of businesses. Image analysis necessitates determining the number of objects in an image. Object counting is a technique for extracting a specific number of items from photographs. These elements serve as a data source for quantitative, motion tracking, and qualitative analysis. The traditional approach for counting objects is manual, time-consuming, and non-automatic. Counting continuously causes eye fatigue and reduces the accuracy of the results. However, counting things is not always simple or easy, especially when done manually. The majority of counting procedures include quirks that make them difficult to master. Despite the fact that object detection has attracted a lot of attention and has made significant progress with deep learning techniques in recent years, these algorithms are not always the best for dealing with sequences of images captured by drone-based platforms due to issues like viewpoint change, scales, and occlusion. Various algorithms have been used in computer vision to count objects, particularly for estimating the number of people in crowded environments.

Most state-of-the-art algorithms are based on Convolutional Neural Networks (CNNs) because they are more resilient to the common difficulties we face, such as varying sizes and perspectives, inter-object occlusions, non-uniform illumination of the scene, and so on. In recent years many versatile object detection algorithms have been proposed like Fast R-CNN [22]. The major drawback associated with the authors' proposed framework is when the scene is highly cluttered and targets seldom split from one another, the tracker may fail to combine all of the targets or dismiss them as false alarms. Further, when a target's motion cannot be accurately described by a constant velocity model, this solution favors dividing the track, although the appearance remains consistent.

The authors in [23] propose an object-based bi-directional counting system that includes enhanced object recognition and tracking algorithm for counting the people flow in the monitoring scene. The authors have used advanced features of CNN to construct their object counting model. The problem associated with the authors' approach is they have used minimal real-time scenarios to evaluate their proposed object-counting model. Hence, further modifications and improvements are needed to convert the proposed model into a generic model.

Object detection algorithms based on deep learning methods completely outperformed others. They can be divided into two main categories, the one-stage approach, and the two-stage approach [24–26]. The one-stage approach uses a fixed number of predictions on a grid to define the bounding boxes around the object, then it tries to classify each bounding box and map it into one of the different classes defined by the neural network. YOLO (You only look once) [27–29], and SSD (Single Shot Detector) [30,31] are one-stage approach detection algorithms that are highly prioritized. The two-stage approaches calculate the position and size of the bounding boxes using a neural network before classification. Therefore, the two-stage approach is more accurate, but it involves more computational power because two neural networks are implemented, one for forming the bounding boxes, and one for classification. R-CNN (Region-based Convolutional Neural Networks) is a two-stage approach algorithm that is widely used. There are two other versions of R-CNN, Faster R-CNN, and Mask R-CNN [32]. Object detection and tracking are done based on the specified categories. The first two categories are for object detection and the last two

are for object tracking. The first category is object detection in images, where the job is to detect objects of a given set of object classes (e.g., vehicles and pedestrians) from individual photographs obtained by any source. In the second category, objects are detected from the videos collected from various sources. Object tracking is done based on two methods, the first method is single tracking where the goal of the work is to estimate the condition of a target, which is indicated in the first frame, in real-time across frames. The second method is muti object tracking where the goal of the task is to recover object trajectories in each video frame using prior detection or without prior detection. Object tracking algorithms are extremely useful in many applications, they can be divided into four main categories [33], matching-based tracking; filtering-based tracking, class-based tracking, and fusion-based tracking. More recently, several deep learning methods are combined with other algorithms to improve tracking performance [34].

Object Counting based on deep learning methods is an active research area that attracted many scholars in recent years. The authors in [35] developed object counting algorithms to count vehicles using the Single Shot Detector (SSD) algorithm. Again, the cost to establish the infrastructure and train the model is complicated and expensive. Further test time detection associated with the model is slow. Some tried to establish an object counting quantitative comparison between background subtraction, Viola-Jones, and Deep Learning Methods on four different datasets [36]. Others investigated YOLO's potential in object counting. One category of researchers build a traffic counting system based on YOLO. They have utilized simple distance calculations to achieve the purpose of vehicle counting, and they added checkpoints to alleviate the consequence of false detection [37,38].

Similarly, other researchers used YOLO as a primary object detector, and they have combined it with correlation filters to build an object counting system [39]. Correlations filters correlate two samples to find the similarity between them. Once an object is detected, they start tracking it until it gets out of the frame or disappears. However, this is a computationally expensive algorithm since it tracks every object in the frame. Moreover, it is more vulnerable to tracking failures as objects that disappear and reappear might be counted twice. Finally, to avoid counting the same object twice, and to reduce the computational complexity, ref. [40] suggested detecting objects every N frame, then using Kanade–Lucas–Tomasi feature tracker (KLT) to track counted objects.

I.C. Amita et al. [41] introduced a novel intelligent real-time traffic monitoring system. A filtered You Only Look Once (YOLO) is employed to achieve vehicle detection. The YOLO framework has been pre-trained to recognize 80 items. The suggested technology is being tested on three different types of vehicles: buses, trucks, and cars. After extracting the three groups, a checkpoint is assigned to determine the number of vehicles in each lane. The count is utilized to manage the road traffic signal in real-time. Three separate publicly available traffic movies are used to evaluate the system. The authors have used Kernel Correlation Filter (KCF) along with the YOLO model for object detection and counting. The inclusion of the KCF filter has improved the system's accuracy. However, the model misclassified between the trucks and trains, which affected the accuracy of object detection and counting. (Object Detection Using YOLO Framework for Intelligent Traffic Monitoring).

Nguyen Duy Noi et al. [42] have provided a method for detecting and counting vehicles in mixed traffic flows using YOLOv4 in the context of Vietnamese transportation. Motorcycles, bicycles, automobiles, trucks, and buses were among the five vehicle types tested on the network. The test results reveal that the authors' algorithm outperforms others (such as Background Subtraction and Haar Cascade) in terms of vehicle detection accuracy and counting on the test set, showing that the suggested technique has better detection performance. However, the authors have used limited traffic scenarios to evaluate their model, and they have to test their model's efficiency for larger diversified datasets. Similarly, Mohammed A. A. Al-qaness [43] has proposed a vehicle tracking system based on intelligent video surveillance. To monitor automobiles, the suggested system employs a neural network, image-based tracking, and YOLOv3 (You Only Look Once). With diverse

datasets, they train the suggested system. Moreover, the authors tested the proposed system's performance using real-world video sequences of traffic. The results of the study revealed that the suggested system can identify, track, and count cars in a variety of situations with acceptable results.

Most of the models discussed in the literature exhibit the following pitfalls. Sometimes when the environment information is not clear, the data stored in the image frame gets scattered distorting object information. The tracker tries to accumulate the scattered information into a single piece of information. This process might decrease the accuracy of classification or the models dismiss the information as false alarms. In some cases, the models misclassify the objects leading to inaccurate environment information. Object counting models, which use ML approaches take more time to train the models. Moreover, more cost is involved to establish the appropriate infrastructure to implement CNN-based models. Some models use small datasets in simulated environments to evaluate their performance. This practice hinders estimating the actual performance and efficiency of the object counting and detecting models.

In the proposed system, we incorporated customized YOLOv5, which has more accuracy compared to previous versions. Most of the authors in the literature used earlier versions of YOLO. We introduce the feature of the crossing line, where we track counted objects for a short period. This approach reduces the computational power required to operate the proposed system; hence, it can be deployed in mobile phones or UAVs. Tracking only counted objects addresses two main causes of tracking failures namely, the increased tracking time which might result in object loss, and an increase in the error rate of the object to template assignment due to instant tracking of multiple objects.

4. System Architecture

Sequentially, the proposed system receives the incoming stream of frames from the UAV and inputs them to the YOLO model. The proposed system contains two main components, the object detection component, and the object counting component. In object detection, we utilize YOLO to detect objects within each frame. We filter out all other objects that belong to unwanted classes. In this project, we are only focusing on vehicles. In the object counting component, we employ CSRT Tracker to track counted objects to avoid counting the same object multiple times. Figure 1 illustrates the High-level Architecture of the system and Figure 2 depicts the workflow of the object counting component.

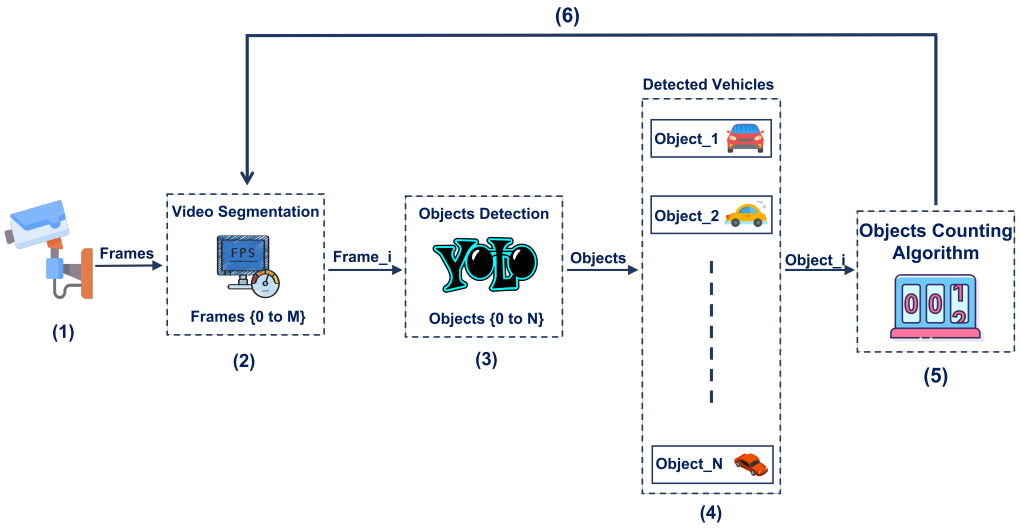


Figure 1. System diagram of the proposed object counting system.

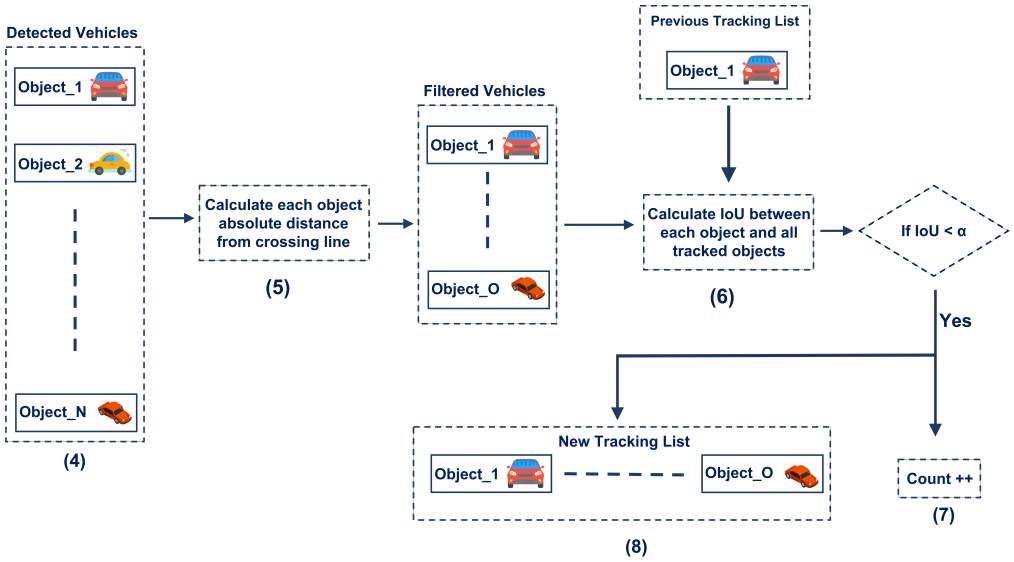


Figure 2. Workflow of Object Counting Algorithm.

4.1. Object Detection Stage

Before executing the YOLO model, positioning the crossing line is mandatory. Objects are counted by the counting algorithm if they cross the line. The position of the line impacts the accuracy of the object counting. Figure 3 shows two different positions of the crossing line. Among the position of the red and yellow lines in the frame, the accuracy of the detected objects is more within the boundary of the red line over the yellow line. If large objects like trucks pass in the center lane within the region of the yellow line, objects passing in the most left lane are not detected by the object detection algorithm. In our system, we choose the position of the crossing line manually by inspection. From the analysis, the study identified that the optimal position for the crossing line must be within the center of the frame as YOLO tends to be more accurate. Each frame is processed by YOLO to find the bounding boxes for all objects in the current frame. Since the YOLO model is trained with many objects, the model can identify and detect multiple classes of objects. The proposed model operates in two modes. In the first mode, the model uses the pre-trained neural network to identify multiple classes, and in the second mode, the model is re-trained to detect the classes of interest. The first mode of operation is a generalized approach, while the second mode focuses on vehicles which improve the accuracy in detecting vehicles. After object detection, the detected objects are forwarded to the object counting module.

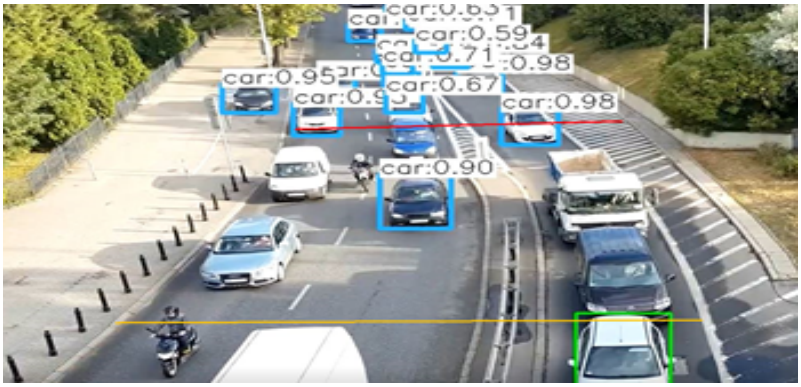


Figure 3. Object detection accuracy changes if position of the crossing line is changed.

4.2. Object Counting Stage

The object counting mechanism is depicted in Figure 2. In this stage, the absolute distance between every individual object and the crossing line is estimated for all the

detected objects. If the distance between an object and the crossing line is less than a threshold τ, and if the object is not in the tracked objects list, the system starts to track the object using the CSRT tracker. Then, the system increments the objects count and adds the identified object to the tracked objects list. The system continuously updates the state of each tracked object using the CSRT tracker. Once an object is added to the tracked objects list, it is is covered by a new bounding box (green colored box) as illustrated in Figure 3. The system keeps tracking each tracked object until it moves away from the crossing line by a threshold of τ. If an object moves away from the crossing line by more than the predefined threshold, the system discards it. The study has initialized the threshold value of τ by trial and error methods, therefore, the value might fluctuate between different street environments. In our system, the threshold value is 30-pixel values.

To make sure that each object is counted once, the system tracks objects until they move away from the crossing line by the predefined threshold. The intersection over union (IoU) is defined by the Equation shown in Figure 4. between the YOLO bounding box and the CSRT tracker bounding box is calculated. The IoU value is used to verify whether the system has counted the identified object. If the IoU between newly detected objects and any tracked object in the tracked objects list is less than a margin μ, then the system starts to track it and increment the object counts.

Some objects might stand within a distance less than the predefined threshold for more than one frame due to congestion or other reasons, due to which redundancy in counting objects occurs. To overcome this issue, this study integrates IoU concepts in the proposed system. Each counted object must be known to the system so that it is counted once. The only possible way is to track every counted object until it moves away from the crossing line by the threshold value.

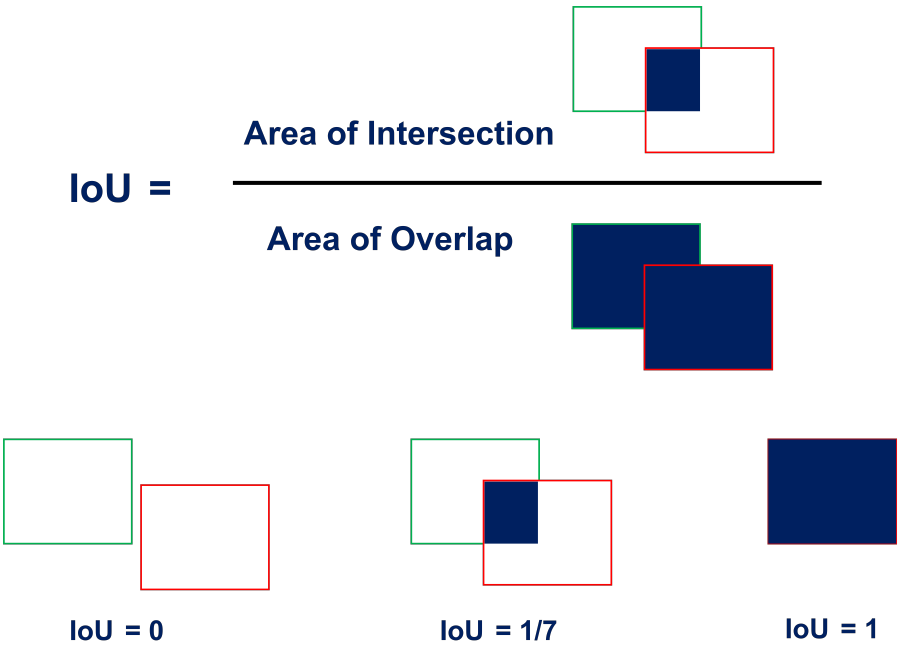


Figure 4. Intersection over Union.

The proposed counting algorithm is efficient over other counting algorithms in terms of less computational power since the system takes less time to track the objects. Instead of tracking the object from the entry and exit points of the frame, the proposed system tracks the object until it moves away from the crossing line by the predefined threshold. Since the probability of tracking failure is less when the tracking region is small, the proposed algorithm has fewer tracking failures over other algorithms that track the object within the entire frame. There won't be many changes in the background environment of the frame if the system tracks the object using a minimum number of frames. If the system

continuously removes objects from the tracked objects list, the CSRT tracker will have minimum templates for searching which improves the tracking accuracy. The pseudocode of the proposed object-counting algorithm is shown in Algorithm 1.

Algorithm 1: Object Counting Algorithm V2

```
Input: frame, crossingLine
Output: counts
trackedObjects = []
counts = 0, threshold = 20, IoUMin = 0.2
detectedObjectes = YOLO(frame)
trackedObjects = CSRT.updateStatus(trackedObjects)
for object in trackedObjects do
    displacement = findDistance(object.position, crossingLine)
    if absolute(displacement) ≥ (threshold) then
        trackedObjects.remove(object)
    else
for object in detectedObjectes do
    displacement = findDistance(object.position, crossingLine)
    if absolute(displacement) < threshold then
        isNewObject = perfromIOU(object, trackedObjects, IoUMin)
        if isNewObject then
            trackedObjects.add(object)
            counts = counts + 1
        else
    else
return counts
```

5. Preparation of Datasets

To test the proposed algorithm for object counting, we have prepared a dataset that consists of 16 videos from different sources. We have chosen the videos so that we have a diversified set of environments and scenarios. The four main parameters chosen to reflect this diversity are object density, image quality, angle of view, and speed of motion. The dataset containing the 16 videos is summarized in Table A1.

5.1. Density of Objects

The object counting algorithm's performance changes according to the number of objects in the frame, therefore, the density of objects is one of the key factors to evaluate the proposed algorithm. The dataset has several videos that have high objects density, medium (normal) objects density, and low objects density. The YOLO performance degrades slightly with the scenes that have a large number of objects, as YOLO misses some of the objects. Moreover, the CSRT tracker will have difficult times finding the tracked objects as the number of similar objects increases with high-density scenes.

5.2. Quality of Images

The proposed algorithm performance degrades with low-resolution images. The dataset has a combination of high, medium, and low-resolution images to evaluate the performance of the proposed system. Further, the study collects data from different time intervals representing a day starting from the early morning till midnight.

5.3. Angle of View

The third parameter is the angle by which the camera is installed, it considers the angle between the camera and the tracked objects. This parameter is important since the

accuracy of YOLO varies with the angle it perceives the object. The system might fail to capture some objects if the cameras are not mounted properly.

5.4. Speed and Direction of Motion

Finally, the speed of motion is also considered during data collection since this feature indirectly affects the object density. If objects are moving fastly, the probability of congestion is low, and the number of frames for tracking the objects is minimum.

6. Performance Evaluation

This section discusses the performance evaluation of the proposed system and the influence of the predefined parameters on the accuracy.

6.1. Density of Objects

Based on the test results summarized in Table A1, it is evident that accuracy is not influenced by the object density feature. The main reason behind this inference is the selection of the appropriate position of the crossing line. The proposed system detects objects in the entire frame, but it tracks the counted objects for the predefined region only. Though the video frame has a large number of objects, the algorithm selects only the objects within the search region. The number of the objects in the search region is at the most equal to the number of road lanes because the algorithm is continuously updating the tracked objects list by removing old objects. In rare cases, YOLO might visualize a single object as dual objects illustrated in the left image Figure 5. However, this is an object detection problem, and cannot be linked to a high object density environment. The percentage of accuracy remains the same in both high-density and low-density environments. In the dataset, video number 10 depicts the low-density environment, and video number 12 illustrates the high-density environment. The difference in the accuracy between video 10 and video 12 is negligible and the accuracy achieved by both is more than 90%.

6.2. Quality of Images

The image quality or the frame resolution is the main cause of performance degradation and higher error rates. The low-resolution frames affect the YOLO and CSRT tracker simultaneously. Hence, the accuracy of the proposed algorithm will drop with low-resolution images. These low-resolution frames cause the YOLO to miss more objects and the system will not be able to count the objects accurately. Even if the YOLO can detect objects in low-resolution environments, CSRT might fail to do the same, resulting in the less accurate object count, as illustrated in the right image of Figure 5. The low image quality affects the accuracy by two means, either YOLO is not detecting the object in the first place, or by the CSRT tracker failing to track. Video number 15 displays a low-resolution and complex environment, the accuracy result is the worst in the whole dataset.

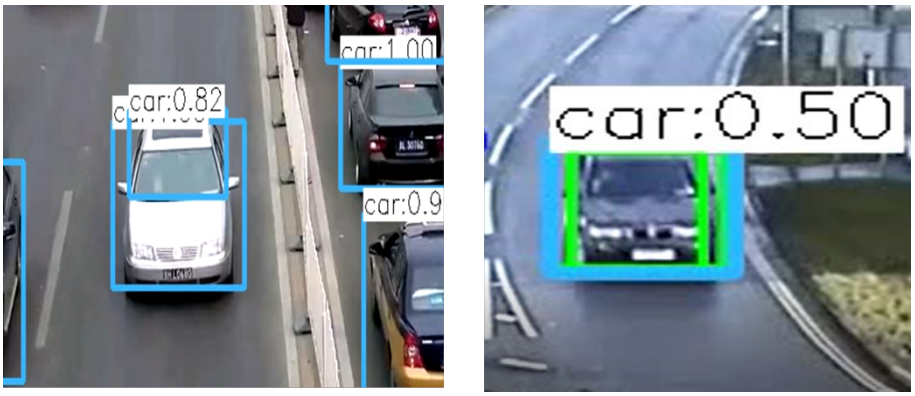


Figure 5. Object density (**Left**) and Image Quality (**Right**).

6.3. Angle of View

The view angle is one of the most important factors that determine the accuracy of the proposed algorithm. A slightly tilted view as illustrated in the left image of Figure 6 will interrupt the vision of the vehicles traveling in other lanes (blind spot detection). We conclude that the camera position has an impact on object detection accuracy. The left image of Figure 6 is an example of the blind spot problem.

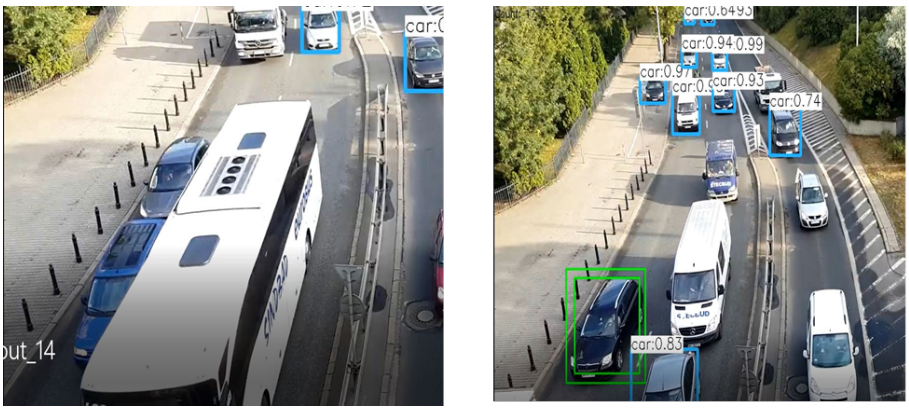


Figure 6. View angle of the camera and the blind spot problem.

6.4. Speed and Direction of Motion

High-speed object motion has a marginal impact on the accuracy of the proposed algorithm. Video number 3, and video number 12 are examples of high-speed environments, with accuracy above 90%. Video number 9 and video number 15 depict less accuracy. The reason behind tracking failure in the CSRT tracker is due to the presence of more objects which move slowly. In low-speed environments, CSRT tracks each counted object for a long duration, compared to high-speed environments. The right image of Figure 6 exhibits tracking failures, which are caused by the redundancy in counting similar objects due to low-speed motion. From the results displayed in Figure 7, it is evident that the inclusion of CSRT tracker YOLO version 5 has improved the accuracy of the object counting over the KCF tracker and YOLO version 2 used in version 1.0 [9].

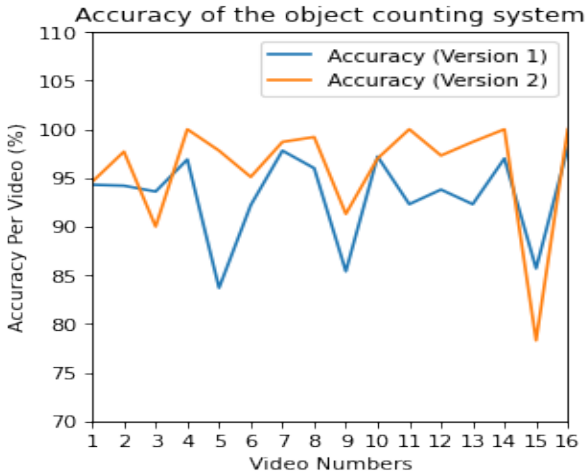


Figure 7. Accuracy comparison between version 1.0 and version 2.0.

6.5. Performance Comparison with Other Related Works

We compared our proposed object counting model with two similar approach models from the existing literature [5,43]. The referred approaches produced a counting accuracy of 86.7% and 85% respectively. Whereas our proposed approach yields 96% average accuracy in counting objects under similar environments.

7. Discussion and Future Work

Object detection and tracking will undoubtedly play a crucial role in creating contextual awareness and perception of autonomous driving for higher levels of autonomy. We plan to extend the contribution of this work in the following directions:

- Implementing the 3GPP Enhanced Sensors use case scenario by evolving the contribution of this article. One of the major ingredients in this case will involve additional sensors to be deployed on the drone. This activity will allow us to investigate the role of 5G C-V2X capabilities (including side-ling, PC 5), etc. for meeting the ITS service quality requirements.
- Investigating the potential of Federated Learning for achieving the objectives of level 5 autonomous driving. For this, we plan to deploy the learning instances at the three levels i.e., vehicle, smart edges (roadside units), and backends of the OEMs, mobile network providers, and ITS service providers. It should be highlighted that the envisioned ecosystem with the engagement of aforementioned stakeholders, federated learning is expected to be a natural fit.
- We plan to evolve the contributed approach to further classify vehicles based on their types and cluster them along with their counts, which can further be analyzed for the use-cases of level 5 autonomous driving.
- Owing to the fact that contribution aligns well with other verticals and application domains, we also plan to extend the proposed methodology in the field of agriculture, to provide detailed analysis on various factors related to plant health status, productivity, and disease attacks. This analysis will guide the agriculturist to improve crop productivity and take early preventive measures on disease attacks.

8. Conclusions

This paper presented an efficient object tracking and counting system for intelligent transportation systems. The proposed system is implemented in two stages. The first stage used the latest YOLO deep learning model (YOLO5) for object detection. In the second stage. we integrated the YOLO5 model with the Channel and Spatial Reliability Tracker (CSRT) to track and count objects within a narrow region to avoid counting the same objects multiple times. To evaluate the performance of the proposed system, we developed a dataset of 20 different videos with various characteristics resembling different image qualities, angle of view, object density, and speed of motion. Performance results revealed that the proposed system could identify and count objects with high accuracy in different scenarios, better than that obtained by its previous version which used the Kernelised Correlation (KCF) Filter.

Author Contributions: All Authors contributed to this work. Conceptualization, A.D. and H.E.; Data curation, M.A., A.M. and M.A.-G.; Formal analysis, A.D. and M.A.-G.; Methodology, A.D., M.A. and A.M.; Supervision, H.E.; Validation, M.A., A.M. and M.A.-G.; Writing—review & editing, A.D., H.I., H.E. and M.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Research Office at the United Arab Emirates University (grant number G00003133) and the Roadway, Transportation, and Traffic Safety Research Center (RTTSRC) of the United Arab Emirates University (grant number 31R116).

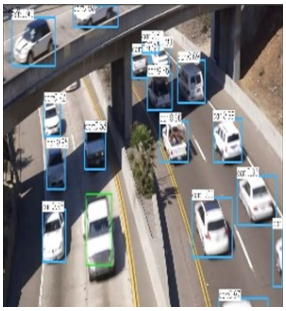
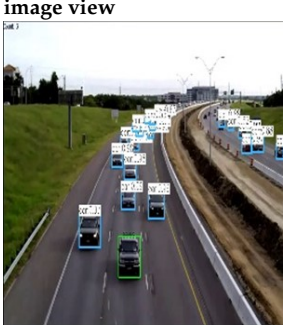
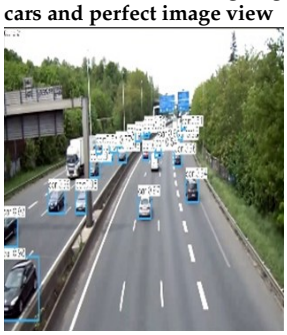
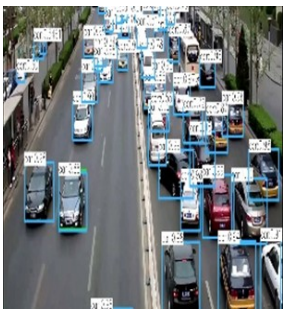
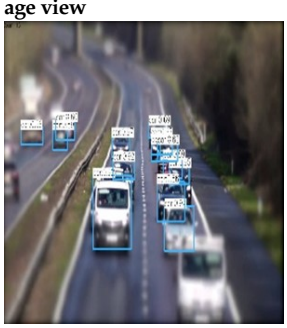
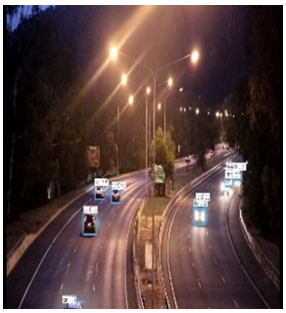
Data Availability Statement: Not applicable, the study does not report any data.

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

Appendix A

This appendix includes the table with a brief description of each video used in the system testing, an image showing the environment the video is representing, and the accuracy of the proposed system under such environment.

Table A1. Dataset used in evaluation of the system performance.

Normal traffic with trucks and perfect image view	**Normal car density with buses and perfect image view**	**High car density with high speed and curved road**	**Aerial image view with traffic density and good image resolution**
			
version[1.0] = 94.3% **version[2.0] = 94.6%**	**version[1.0] = 94.2%** **version[2.0] = 97.7%**	**version[1.0] = 93.6%** **version[2.0] = 90.0%**	**version[1.0] = 96.9%** **version[2.0] = 100%**
Curved road with normal traffic and blind spot	**Normal traffic and perfect image view**	**Normal traffic with trucks and blind spot**	**Normal traffic with outgoing cars and perfect image view**
			
version[1.0] = 83.7% **version[2.0] = 97.8%**	**version[1.0] = 92.2%** **version[2.0] = 95.1%**	**version[1.0] = 97.8%** **version[2.0] = 98.7%**	**version[1.0] = 96.0%** **version[2.0] = 99.2%**
High traffic density with low speed	**Low traffic density and perfect image view**	**Normal traffic and perfect image view**	**High traffic density and speed with occlusion and perfect image view**
			
version[1.0] = 85.4% **version[2.0] = 91.3%**	**version[1.0] = 97.2%** **version[2.0] = 97.0%**	**version[1.0] = 92.3%** **version[2.0] = 100%**	**version[1.0] = 93.8%** **version[2.0] = 97.3%**
Normal traffic with trucks and blind spot	**Normal traffic at night**	**High traffic with distraction and noise and low speed**	**Extra Low traffic and perfect image view**
			
version[1.0] = 92.3% **version[2.0] = 98.7%**	**version[1.0] = 97.0%** **version[2.0] = 100%**	**version[1.0] = 85.7%** **version[2.0] = 78.2%**	**version[1.0] = 98.1%** **version[2.0] = 100%**

References

1. Kugler, L. Real-world applications for drones. *Commun. ACM*. **2019**, *62*, 19–21. [CrossRef]
2. Rosser, J.C., Jr.; Vignesh, V.; Terwilliger, B.A.; Parker, B.C. Surgical and medical applications of drones: A comprehensive review. *J. Soc. Laparoendosc. Surg.* **2018**, *22*. Available online: https://pubmed.ncbi.nlm.nih.gov/30356360/ (accessed on 27 November 2021). [CrossRef] [PubMed]
3. Hassanalian, M.; Abdelkefi, A. Classifications, applications, and design challenges of drones: A review. *Prog. Aerosp. Sci.* **2017**, *91*, 99–131. [CrossRef]
4. Hassanalian, M.; Rice, D.; Abdelkefi, A. Evolution of space drones for planetary exploration: A review. *Prog. Aerosp. Sci.* **2018**, *97*, 61–105. [CrossRef]
5. Mirthubashini, J.; Santhi, V. Video Based Vehicle Counting Using Deep Learning Algorithms. In Proceedings of the 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, India, 6–7 March 2020; pp. 142–147. [CrossRef]
6. Dai, Z.; Song, H.; Wang, X.; Fang, Y.; Yun, X.; Zhang, Z.; Li, H. Video-Based Vehicle Counting Framework. *IEEE Access* **2019**, *7*, 64460–64470. [CrossRef]
7. Yang, Y.; Gao, W. A Method of Pedestrians Counting Based on Deep Learning. In Proceedings of the 2019 3rd International Conference on Electronic Information Technology and Computer Engineering (EITCE), Xiamen, China, 18–20 October 2019; pp. 2010–2013. [CrossRef]
8. Mamdouh, N.; Khattab, A. YOLO-Based Deep Learning Framework for Olive Fruit Fly Detection and Counting. *IEEE Access* **2021**, *9*, 84252–84262. Available online: https://ieeexplore.ieee.org/abstract/document/9450822 (accessed on 27 November 2021). [CrossRef]
9. Dirir, A.; Adib, M.; Mahmoud, A.; Al-Gunaid, M.; El-Sayed, H. An Efficient Multi-Object Tracking and Counting Framework Using Video Streaming in Urban Vehicular Environments. In Proceedings of the 2020 International Conference on Communications, Signal Processing, and their Applications (ICCSPA), Sharjah, United Arab Emirates, 16–18 March 2021; pp. 1–7. [CrossRef]
10. Farkhodov, K.; Lee, S.-H.; Kwon, K.-R. Object Tracking Using CSRT Tracker and RCNN Farkhodov2020object. 2020, pp. 209–212. Available online: https://www.scitepress.org/Papers/2020/91838/91838.pdf (accessed on 27 November 2021).
11. Piccardi, M. Background subtraction techniques: A review. *IEEE Proc. Int. Conf. Syst. Man Cybern.* **2004**, *4*, 3099–3104.
12. Del Blanco, C.R.; Jaureguizar, F.; García, N. Visual tracking of multiple interacting objects through raoblackwellized data association particle filtering. In Proceedings of the 2010 IEEE International Conference on Image Processing, Hong Kong, China, 26–29 September 2010; pp. 821–824. Available online: https://ieeexplore.ieee.org/document/5653411 (accessed on 27 November 2021).
13. Genovesio, A.; Olivo-Marin, J.C. Split and merge data association filter for dense multi-target tracking. In Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004, Cambridge, UK, 26 August 2004; Volume 4, pp. 677–680.
14. Ma, Y.; Yu, Q.; Cohen, I. Target tracking with incomplete detection. *Comp. Vis. Image Underst.* **2009**, *113*, 580–587. [CrossRef]
15. Yu, Q.; Medioni, G. Multiple-target tracking by spatiotemporal monte carlo markov chain data association. *IEEE Trans. Patern Anal. Mach. Intell.* **2009**, *31*, 2196–2210.
16. Khan, Z.; Balch, T.; Dellaert, F. Multitarget tracking with split and merged measurements. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 20–25 June 2005; Volume 1, pp. 605–610.
17. Del Blanco, C.R.; Jaureguizar, F.; García, N. Bayesian Visual Surveillance: A Model for Detecting and Tracking a variable number of moving objects. In Proceedings of the 2011 18th IEEE International Conference on Image Processing, Brussels, Belgium, 11–14 September 2011; pp. 1437–1440. Available online: https://ieeexplore.ieee.org/document/6115712 (accessed on 27 November 2021).
18. Yam, K.; Siu, W.; Law, N.; Chan, C. Effective bidirectional people flow counting for real time surveillance system. In Proceedings of the 2011 IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, USA, 9–12 January 2011; pp. 863–864.
19. Tuzel, O.; Porikli, F.; Meer, P. Learning on lie groups for invariant detection and tracking. In Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition, Anchorage, AK, USA, 23–28 June 2008; pp. 1–8. Available online: https://ieeexplore.ieee.org/document/4587521 (accessed on 27 November 2021).
20. Galvez, R.L.; Bandala, A.A.; Dadios, E.P.; Vicerra, R.R.; Maningo, J.M. Object Detection Using Convolutional Neural Networks. In Proceedings of the TENCON 2018—2018 IEEE Region 10 Conference, Jeju, Korea, 28 October 2018; pp. 2023–2027. Available online: https://ieeexplore.ieee.org/abstract/document/8650517 (accessed on 27 November 2021).
21. Huang, Z.; Wang, J.; Fu, X.; Yu, T.; Guo, Y.; Wang, R. DC-SPP-YOLO: Dense connection and spatial pyramid pooling based YOLO for object detection. *Inf. Sci.* **2020**, *522*, 241–258. [CrossRef]
22. Sun, X.; Wu, P.; Hoi, S.C. Face detection using deep learning: An improved faster RCNN approach. *Neurocomputing* **2018**, *299*, 42–50. [CrossRef]
23. Broad, A.; Jones, M.; Lee, T.Y. Recurrent Multi-frame Single Shot Detector for Video Object Detection. In BMVC 2018 Sep. p. 94. Available online: http://bmvc2018.org/contents/papers/0309.pdf (accessed on 27 November 2021).
24. Naik, Z.K.; Gandhi, M.R. A Review: Object Detection using Deep Learning. *Int. J. Comput. Appl.* **2018**, *180*, 46–48.

25. Anusha, C.; Avadhani, P. S. Object Detection using Deep Learning. *Int. J. Comput. Appl.* **2018**, *182*, 18–22. [CrossRef]
26. Pathak, A.; Pandey, M.; Rautaray, S. Application of Deep Learning for Object Detection. *Procedia Comput. Sci.* **2018**, *132*, 1706–1717. [CrossRef]
27. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. *Proc. Cvpr* **2016**, 779–788. Available online: https://www.cv-foundation.org/openaccess/content_cvpr_2016/html/Redmon_You_Only_Look_CVPR_2016_paper.html (accessed on 27 November 2021).
28. Redmon, J.; Farhadi, A. Yolo9000: Better, faster, stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 6517–6525.
29. Redmon, J.; Farhadi, A. Yolov3: An incremental improvement. *arXiv* **2018**. arXiv:1804.02767.
30. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; Berg, A.C. Ssd: Single shot multibox detector. In *European Conference on Computer Vision*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 21–37. Available online: https://link.springer.com/chapter/10.1007/978-3-319-46448-0_2 (accessed on 27 November 2021).
31. Fu, C.-Y.; Liu, W.; Ranga, A.; Tyagi, A.; Berg.Dssd: Deconvolutional single shots detector. *arXiv* **2017**, arXiv:1701.06659.
32. He, K.; Gkioxari, G.; Dollar, P.; Girshick, R. Mask R-CNN. In *Proceedings of ICCV*; 2017; pp. 2961–2969. Available online: https://openaccess.thecvf.com/content_iccv_2017/html/He_Mask_R-CNN_ICCV_2017_paper.html (accessed on 27 November 2021).
33. Sunkara, J.; Santhosh, M.; Cherukuri, S.; Gopi Krishna, L. Object Tracking Techniques and Performance Measures—A Conceptual Survey. In Proceedings of the IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI-2017), Chennai, India, 21–22 Sepember 2017.
34. Zhang, D; Maei, H.; Wang, X.; Wang, Y. Deep reinforcement learning for visual object tracking in videos. *Comput. Res. Repos.* **2017**. Available online: https://arxiv.org/pdf/1701.08936.pdf (accessed on 27 November 2021).
35. Saxena, G.; Gupta, A.; Verma, D.K.; Rajan, A.; Rawat, A. Robust Algorithms for Counting and Detection of Moving Vehicles using Deep Learning. In Proceedings of the 2019 IEEE 9th International Conference on Advanced Computing (IACC), Tiruchirappalli, India, 13–14 December 2019; pp. 235–239. [CrossRef]
36. Hardjono, B.; Tjahyadi, H. Vehicle Counting Quantitative Comparison Using Background Subtraction, Viola Jones and Deep Learning Methods. In Proceedings of the 2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON). Available online: https://ieeexplore.ieee.org/abstract/document/8615085 (accessed on 27 November 2021).
37. Lin, J.-P.; Sun, M.-T. A YOLO-Based Traffic Counting System. In Proceedings of the 2018 Conference on Technologies and Applications of Artificial Intelligence (TAAI), Taichung, Taiwan, 30 November–2 December 2018.
38. Asha, C. S.; Narasimhadhan, A.V. Vehicle Counting for Traffic Management System using YOLO and Correlation Filter. In Proceedings of the 2018 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT). Available online: https://ieeexplore.ieee.org/abstract/document/8482380 (accessed on 27 November 2021).
39. Forero, A.; Calderon, F. Vehicle and pedestrian video-tracking with classification based on deep convolutional neural networks. In Proceedings of the 2019 XXII Symposium on Image, Signal Processing and Artificial Vision (STSIVA), Bucaramanga, Colombia, 24–26 April 2019.
40. Mohamed, A.A. Accurate Vehicle Counting Approach Based on Deep Neural Networks. In Proceedings of the 2019 International Conference on Innovative Trends in Computer Engineering (ITCE), Aswan, Egypt, 8–9 February 2020.
41. Amitha, I.C.; Narayanan, N.K. Object Detection Using YOLO Framework for Intelligent Traffic Monitoring. In *Machine Vision and Augmented Intelligence—Theory and Applications*; Springer: Singapore, 2021; pp. 405–412.
42. Can, V.X.; Vu, P.X.; Rui-fang, M.; Thuat, V.T.; Van Duy, V.; Noi, N.D. Vehicle Detection and Counting Under Mixed Traffic Conditions in Vietnam Using Yolov4. Available online: https://d1wqtxts1xzle7.cloudfront.net/67640687/IJARET_12_02_072.pdf?1623818829=&response-content-disposition=inline%3B+filename%3DVEHICLE_DETECTION_AND_COUNTING_UNDER_MIX.pdf&Expires=1638195611&Signature=A7yVUQcYKePsOgUZFH4zqVXbmsP0QpVRlDLAYnmHiCIEDdV6uo4VJS-1T945AeWp~IkEcwak8YlVah0TFMs9mw4rNFO3ISDAFnqciqzKZL2uFZqckHtJIdTSwwJrFDpSk1zgPep6yr8wKQw~6-abIhv-2-yWSOi0DAOzYtFuUzlShv~Z4mWUefyI-OZcxqfDj3SUkaTvELtGCZlrNtXHQa2s0RicIT3xw0mGGFf-6~u-xuaviKnFuCz9~dtn2XCmQEuAkOjCSDn3uQjJksZsA7U4HiUf~ziZ1G9ke~4u~Uv5n6YpAO4KpL0y3oOumdc71J~aHvBE0rzaYtk0rQOO1w__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA (accessed on 27 November 2021).
43. Al-qaness, M.A.; Abbasi, A.A.; Fan, H.; Ibrahim, R.A.; Alsamhi, S.H.; Hawbani, A. An improved YOLO-based road traffic monitoring system. *Computing* **2021**, *103*, 211–230. [CrossRef]