machines

MDPI

Article

KCS-YOLO: An Improved Algorithm for Traffic Light Detection under Low Visibility Conditions

Qinghui Zhou [1], Diyi Zhang [1], Haoshi Liu [1] and Yuping He [2,*]

[1] School of Mechatronics and Vehicle Engineering, Beijing University of Civil Engineering and Architecture, Beijing 100044, China; zhouqinghui@bucea.edu.cn (Q.Z.); 2108550023006@stu.bucea.edu.cn (D.Z.); 2108550021081@stu.bucea.edu.cn (H.L.)

[2] Department of Automotive and Mechatronics Engineering, University of Ontario Institute of Technology, Oshawa, ON L1G 0C5, Canada

* Correspondence: yuping.he@uoit.ca

Abstract: Autonomous vehicles face challenges in small-target detection and, in particular, in accurately identifying traffic lights under low visibility conditions, e.g., fog, rain, and blurred night-time lighting. To address these issues, this paper proposes an improved algorithm, namely KCS-YOLO (you only look once), to increase the accuracy of detecting and recognizing traffic lights under low visibility conditions. First, a comparison was made to assess different YOLO algorithms. The benchmark indicates that the YOLOv5n algorithm achieves the highest mean average precision (mAP) with fewer parameters. To enhance the capability for detecting small targets, the algorithm built upon YOLOv5n, namely KCS-YOLO, was developed using the K-means++ algorithm for clustering marked multi-dimensional target frames, embedding the convolutional block attention module (CBAM) attention mechanism, and constructing a small-target detection layer. Second, an image dataset of traffic lights was generated, which was preprocessed using the dark channel prior dehazing algorithm to enhance the proposed algorithm's recognition capability and robustness. Finally, KCS-YOLO was evaluated through comparison and ablation experiments. The experimental results showed that the mAP of KCS-YOLO reaches 98.87%, an increase of 5.03% over its counterpart of YOLOv5n. This indicates that KCS-YOLO features high accuracy in object detection and recognition, thereby enhancing the capability of traffic light detection and recognition for autonomous vehicles in low visibility conditions.

Keywords: object detection and recognition; YOLOv5n; KCS-YOLO (an improved algorithm); low visibility; autonomous vehicles

Citation: Zhou, Q.; Zhang, D.; Liu, H.; He, Y. KCS-YOLO: An Improved Algorithm for Traffic Light Detection under Low Visibility Conditions. *Machines* **2024**, *12*, 557. https://doi.org/10.3390/machines12080557

Academic Editor: Jiangjian Xiao

Received: 30 June 2024
Revised: 7 August 2024
Accepted: 14 August 2024
Published: 15 August 2024

1. Introduction

Traffic light detection and recognition is important for the safe operation of autonomous vehicles. However, low visibility, as one of the main factors causing traffic accidents, poses a potential safety hazard for intelligent vehicles. Visibility, in the context of transportation, refers to the distance at which one can identify objects ahead under a complex meteorological condition [1]. It is not only an important weather indicator, but also a pivotal factor for ensuring safe transportation [2].

Two main factors affect visibility: (1) illumination difference, and (2) atmospheric transparency. Natural and artificial lights are two factors affecting illumination. The illumination value is directly proportional to the visibility value, implying that higher illumination corresponds to greater visibility. Severe weather conditions, such as heavy rain, snow, fog, and sandstorms, can make the atmosphere turbid [3]. Atmospheric transparency is directly proportional to visibility, and reduced transparency leads to decreased visibility. Generally, visibility below 300 m is considered low visibility, implying a challenging environment for autonomous vehicle systems. A long distance between a traffic light and a small target may diminish the brightness contrast between the object and the background,

making them difficult to be detected. Additionally, adverse weather conditions, such as rain, snow, and smog, significantly reduce atmospheric transparency. These factors, which lead to poor visibility, can detrimentally impact driver vision, making driving more challenging and potentially leading to severe traffic accidents [4] Given the essentiality of traffic light detection in low visibility conditions, this paper proposes an algorithm for enhancing the capabilities of detecting and recognizing traffic lights when visibility is below 300 m. This algorithm will increase the safety of autonomous vehicle operations.

The proposed algorithm, i.e., KCS-YOLO (you only look once), is built upon YOLO-series algorithms, which are devoted to addressing the challenges posed by traffic lights at a long distance and low visibility, caused by rain, snow, and blurred night-time lighting. To develop KCS-YOLO, we initially compared different YOLO algorithms, among which the algorithm (called YOLOv5n) with superior performance was selected. KCS-YOLO was designed to improve the performance of YOLOv5n. KCS-YOLO utilizes the K-means++ algorithm for clustering labeled multi-dimensional bounding boxes, and integrates the convolutional block attention module (CBAM) attention mechanism to adaptively focus on more critical features [5], thereby improving the algorithm performance. Additionally, it incorporated a small object detection layer to improve the accuracy of detecting small traffic light targets. This was followed by preprocessing the generated traffic light image dataset using the dark channel prior dehazing algorithm, which effectively enhanced the visual contrast threshold. Finally, comparison and ablation experiments were conducted among KCS-YOLO, YOLOv5n, its variants, and a popular algorithm, called Faster-RCNN (region-based convolution neural network), to validate the effectiveness of the proposed algorithm.

The remainder of this paper is structured as follows. Section 2 introduces the related studies on traffic light detection. Section 3 describes the generation of the traffic light dataset and the development of the KCS-YOLO algorithm, including the integration of the K-means++ clustering algorithm, the CBAM attention mechanism, and a small object detection layer. Section 4 demonstrates the effectiveness of the proposed algorithm based on the experimental results. Finally, the conclusions are drawn in Section 5.

2. Related Studies

Traditional traffic light detection usually starts with color features and combines machine learning, image processing, and other algorithms for recognition. However, traditional detection algorithms exhibit low accuracy, thereby leading to missed detections and false alarms. With the rapid development of deep learning, object detection algorithms based on computer vision have been widely applied; extensive studies have been conducted to apply deep learning techniques to traffic light detection for autonomous vehicles. Tian and Kim have preprocessed images using the dark channel prior based on the classification result, and then performed vehicle detection based on the preprocessed images using Faster-RCNN [6]. However, the improved algorithm still needs to be improved in both accuracy and real-time performance. Compared with Faster-RCNN, YOLOv3 achieves faster detection, which has attracted significant attention from researchers [7]. Possatti et al. integrated prior map information with YOLOv3 for traffic light detection [8], while Du et al. proposed an improved YOLOv3 algorithm by upsampling operations for multi-feature fusion [9].

Nevertheless, these algorithms still suffered from low detection accuracy. Kou proposed a road traffic sign recognition algorithm, which combines image preprocessing and deep learning neural networks [10]. The algorithm enhanced image attributes, e.g., contrast and brightness, to reduce interference caused by adverse weather conditions. Mondal et al. developed an algorithm, which is built upon a contrast-limited adaptive histogram equalization-based multiscale fusion technique [11]. The above algorithms significantly improved sign detection accuracy using deep learning techniques. As new versions of the YOLO algorithm family continued to emerge, researchers increasingly applied them to traffic light detection tasks. Li et al. proposed a faster and more accurate

small-target detection algorithm for traffic light detection based on YOLOv5 [12]. Subsequently, Mao et al. optimized the YOLOv7 algorithm by introducing attention mechanisms and a SIOU (SCYLLA intersection over union) loss function [13]. Although this algorithm enhanced the traffic light detection capability, it did not consider complex weather conditions. Inclement weather conditions may adversely impact the performance of algorithms for traffic light detection and recognition. To address this issue, Appiah and Mensah designed a hybrid technique, which combines two algorithms, namely YOLOv7 and ESRGAN (enhanced super-resolution generative adversarial networks) [14]. The latter was used to learn from a set of training data and adjust the images under unfavorable weather conditions; subsequently, the former performed object detection. ESRGAN enhanced each image so that YOLOv7 could improve detection accuracy under adverse weather conditions. Li et al. addressed light recognition in challenging environments by integrating traditional techniques, deep learning, prior knowledge, and multi-sensor data with an improved YOLOv4 model [15]. However, this work predates YOLOv5 and its variants, resulting in less optimal accuracy.

The above literature review indicates that it is indispensable to further improve the recognition accuracy under complex conditions. Therefore, this study aims at addressing the problem of low traffic light detection accuracy in low visibility conditions by developing a new algorithm built upon the existing YOLO algorithms, thereby increasing the precision of traffic light detection.

3. Traffic Light Dataset and Proposed KCS-YOLO Algorithm

This section introduces the requested traffic light dataset and the proposed KCS-YOLO algorithm.

3.1. Generating Dataset

We collected 3053 traffic light samples in low visibility conditions, with images filtered and manually generated for our traffic light dataset. The generated dataset covers various time periods, rainy conditions, as well as different intensities of red, yellow, and green lights for different operating scenarios, e.g., straight ahead, left turn, and right turn. Figure 1 shows the sample images from the generated dataset. Given the generated dataset, the images were processed using the operations, e.g., flipping, translation, rotation, cropping, scaling, adding noise, and random occlusion, to enhance the diversity and representativeness. LabelImg was then employed for image annotation, and the custom dataset contained only traffic light color labels, with four categories: “red”, “yellow”, “green” and “off” [16]. The dataset was divided into three parts: the training set comprised 80%, the validation set comprised 10%, and the testing set comprised 10%.

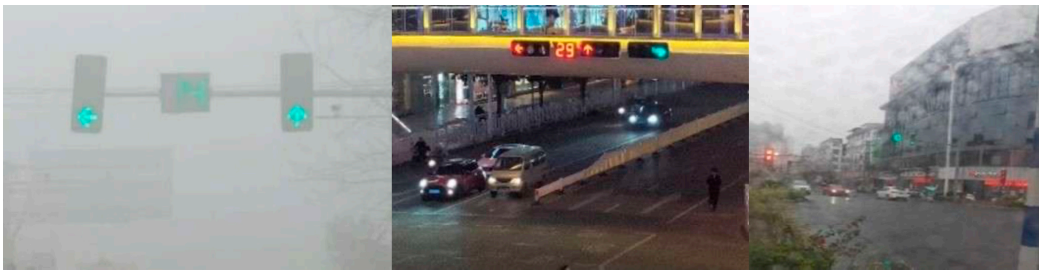

Figure 1. Sample images from generated traffic light dataset.

3.2. Preprocessing Dataset

Due to the influence of atmospheric scattering in low visibility environments, such as foggy weather, the detection and recognition processing for traffic lights may be slower and less accurate. Therefore, this study utilized the dark channel prior dehazing method [17]

for image preprocessing in order to enhance the clarity and obtain clearer dehazed images. In computer vision, for image dehazing, the imaging model of haze is simplified as

$$I(x) = J(x)t(x) + A(1 - t(x)) \tag{1}$$

where $J(x)$ represents the dehazed image, A the global atmospheric light, $I(x)$ the hazy image, and $t(x)$ the transmission rate of light. From the haze imaging model expressed in (1), it is known that the key to obtaining a dehazed image $J(x)$ lies in calculating the transmission rate $t(x)$ and the global atmospheric light value A.

3.2.1. Dark Channel Prior

He et al. experimentally found that for the vast majority of non-sky local regions in an RGB (red, green, and blue) image, there exists a channel with an intensity approaching zero, which is known as the dark channel $J^{dark} \to 0$ [17]. The definition of the dark channel for any image $J(x)$ is represented as

$$J^{dark}(x) = \min_{c \in \{r,g,b\}} \left(\min_{y \in \Omega(x)} (J^c(y) \right) \tag{2}$$

where $\Omega(x)$ denotes a local region centered at x, $J^c(y)$ is the color channel of J, with c representing the RGB channels, and $J^{dark}(x)$ signifies the dark channel. Due to haze, the light scattering increases, the image contrast decreases, and the value of the dark channel no longer tends to zero. Consequently, the value of dark channel in a hazy image can be approximately considered a measure of fog density.

3.2.2. Estimating Transmission Rate

The estimation of transmission rate is the core of the dark channel prior dehazing method. In the foggy imaging model of the dark channel, the transmission rate is assumed as $t(x)$. The global atmospheric value A is treated as a constant, and the dark channel value $J^{dark}(x)$ tends to zero for a clear image. To achieve depth perception in the image, a buffer factor ω (usually set to 0.95) [18] is introduced for correction. The formula for calculating the transmission rate $t(x)$ is given by

$$t(x) = 1 - \omega \min_{c \in \{r,g,b\}} \left\{ \min_{y \in \Omega(x)} \left(\frac{I^c(y)}{A^c} \right) \right\} \tag{3}$$

3.2.3. Image Dehazing

For the dark channel prior dehazing method by He et al. [17], the overall dehazing process can be schematically illustrated, as shown in Figure 2.

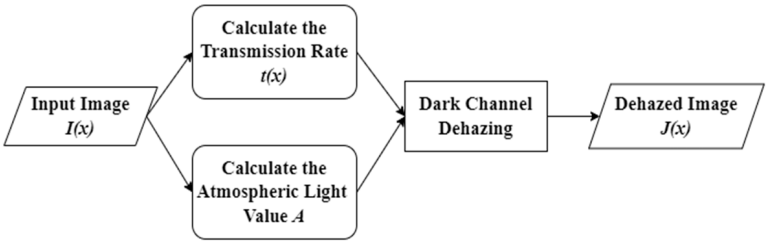

Figure 2. Dehazing flowchart.

When the transmission rate $t(x)$ is too small, it can cause the $J(x)$ value to be excessively large, resulting in overexposure in the dehazed recovery image. Therefore, a lower

bound for the transmittance t_0 is set to 0.1, retaining a small amount of haze. Thus, the formula for the final restored picture can be cast as

$$J(x) = \frac{I(x) - A}{\max[t(x), t_0]} + A \tag{4}$$

The dark channel prior algorithm can be used to improve image quality, and the dehazing effect of a sample image is shown in Figure 3. In addition, this algorithm can preprocess rainy images to enhance clarity and obtain clearer pictures. Raindrops cause scattering in rainy images, thereby leading to decreased image contrast. The dehazing process can enhance the contrast of images, thereby making raindrops and other objects clearer. While restoring the clarity of images, it also maintains the original colors, which helps in restoring the true colors of rainy scenes. Moreover, the dehazing algorithm can eliminate the scattering caused by raindrops, making the image clearer.

(**a**) (**b**)

Figure 3. An example of: (**a**) hazy image; (**b**) dehazing image.

In this research, dehazing preprocessing was carried out before the preprocessed images were fed to KCS-YOLO for traffic light detection. The perception system captures original images directly; then, the images are preprocessed using the dehazing algorithm, and finally, the preprocessed images are exported to the proposed KCS-YOLO algorithm for traffic light detection.

3.3. A Comparison of Different YOLO Algorithms

The first version of the YOLO algorithm, proposed by Redmon et al. in 2015 [19], is a target detection technique based on a single neural network. YOLO, which is based on the concept of regression, shows distinguished advantages, e.g., rapid detection speed, ease of implementation, and end-to-end optimization. Therefore, it has been widely applied to various engineering fields, e.g., autonomous driving, intelligent transportation systems, and robot navigation [20]. Since the first version of YOLO developed in 2015, various variants of the algorithm have been proposed and designed. For simplicity, we treat these variants of the original algorithm as different YOLO algorithms.

Table 1 compares the selected YOLO algorithms in the performance measures of average precision (AP), mean average precision (mAP), parameters (Params/M), F1 score, and floating-point operations per second (FLOPs/G). These metrics are evaluated by

$$P = \frac{TP}{TP + FP} \tag{5}$$

$$R = \frac{TP}{TP + FN} \tag{6}$$

$$AP = \int_0^1 P \cdot R dR \tag{7}$$

$$mAP = \frac{1}{N} \sum_{i=1}^{N} AP_i \tag{8}$$

$$F1 = 2\frac{P \cdot R}{P + R} \tag{9}$$

$$FLOPs = 2HW\left(C_{in}K^2 - 1\right)C_{out} \tag{10}$$

$$Params = C_{in}C_{out}K^2 \tag{11}$$

where P and R represent the precision and recall of the algorithm, respectively, TP (true positive) denotes the number of positive samples correctly predicted as positive, FP the number of negative samples incorrectly predicted as positive, FN the number of positive samples incorrectly predicted as negative, and N the total number of classes in the dataset; H, W and C_{in} are the height, width, and number of channels of the input feature map, correspondingly, K is the kernel width (assumed to be symmetric), and C_{out} the number of output channels.

Table 1. A comparison of different YOLO algorithms.

Algorithms	AP				mAP	F1				Params/M	FLOPs/G
	Red	Green	Yellow	Off		Red	Green	Yellow	Off		
YOLOv8n	0.9663	0.9441	0.9731	0.6750	0.8896	0.95	0.93	0.96	0.50	3.157	8.858
YOLOv7	0.9876	0.9598	0.9336	0.8647	0.9264	0.96	0.95	0.89	0.75	37.620	106.472
YOLOv5n	0.9799	0.9588	0.9227	0.8922	0.9384	0.98	0.95	0.90	0.80	7.277	17.156
YOLOv4	0.9711	0.9435	0.8701	0.5538	0.8346	0.94	0.93	0.81	0.64	64.363	60.527
YOLOv3	0.7548	0.7535	0.7377	0.6694	0.7289	0.77	0.75	0.71	0.64	61.949	66.171

The mAP is an important performance index; it is the mean of average precision over all classes, considering both precision (P) and recall (R); the higher the score, the more accurate the algorithm in object detection. The recall rate denotes the ability to find a true positive among the actual positive samples, while the precision rate mainly reflects the proportion of true positives among the total number predicted as positive. AP is the average of the precision at different recall values under a given threshold. F1 measures the harmonic average of precision and recall, reflecting the stability of the algorithm; the lower the F1 value, the greater the imbalance between precision and recall will be. Params (parameters) denotes the total number of learnable parameters in an algorithm, which is a measure of algorithm complexity and memory footprint. FLOPs represent the number of floating-point operations performed per second by an algorithm during inference. It is a measure of computational complexity and efficiency. It tells you how many "calculations" the algorithm needs to do to understand an image and find objects in it.

The data listed in Table 1 are attained from the experiments on the generated dataset. Figure 4 shows the comparison of these algorithms in mAP, Params/M, Flop/G, and F1. As shown in this table, there exists a large gap between YOLOv8n and YOLOv7 in terms of Params/M and Flop/G. YOLOv8n exhibits better performance in these measures due to the following reason: Within the YOLOv8 series, YOLOv8n strikes a balance between performance and computational efficiency. By means of incorporating novel network architectures, optimizing training strategies and parameters, YOLOv8n is able to maintain high detection accuracy while reducing computation burdens.

Compared with YOLOv8n, YOLOv5n achieves higher detection precision with a similar model size. YOLOv5n is a lightweight algorithm within the YOLOv5 series, and it features a relatively small number of parameters and good detection accuracy. Moreover, this algorithm offers a fast inference speed, making it suitable for scenarios requiring rapid detection, including real-time target detection. Therefore, YOLOv5n, which has the highest detection precision among the algorithms listed in Table 1, a smaller number of algorithm parameters, and the best overall performance, has been selected as the baseline algorithm for further improvement.

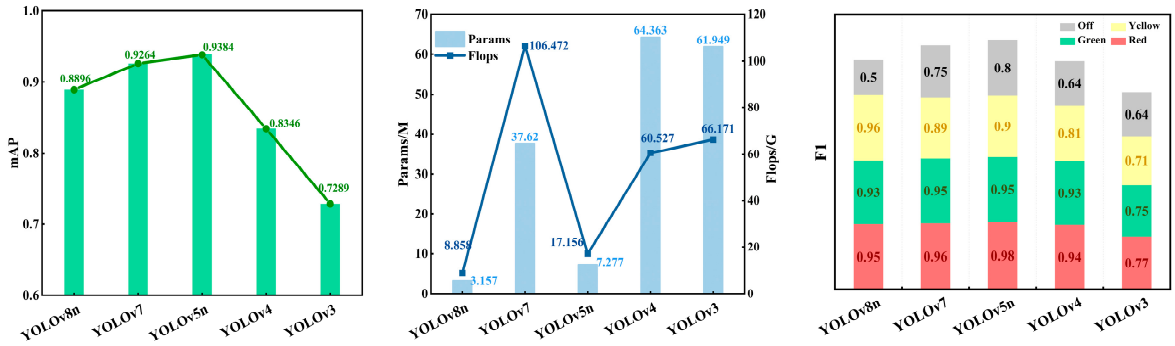

Figure 4. Comparison of performance measures for different YOLO algorithms.

3.4. Proposed KCS-YOLO Algorithm

YOLOv5n algorithm consists of three main components: backbone, neck, and head. It is widely used in real-time target detection applications and achieves good detection performance. However, YOLOv5n algorithm still suffers from slow detection speed and low accuracy in poor weather conditions for detecting traffic lights. Therefore, built upon the YOLOv5n algorithm, KCS-YOLO is developed to enhance the ability of detecting and recognizing traffic lights. The integrated architecture of the proposed KCS-YOLO algorithm is depicted in Figure 5.

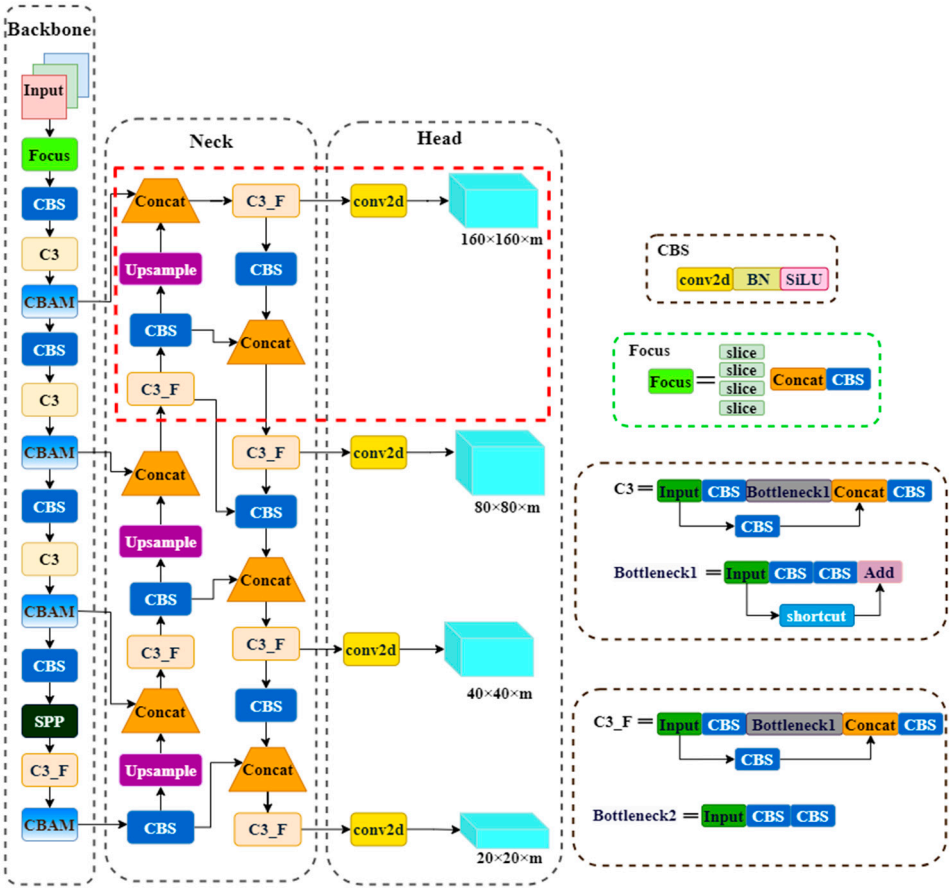

Figure 5. Architecture of KCS-YOLO.

KCS-YOLO algorithm is developed by taking the following measures: (1) using the K-means++ algorithm for clustering marked multi-dimensional target frames, (2) embedding the CBAM attention mechanism, and (3) constructing a small-target detection layer.

3.4.1. K-Means++ Re-Clustering Anchors

The baseline algorithm, i.e., YOLOv5n, was trained using the public available COCO (Common Objects in Context) dataset, employing the K-means clustering algorithm to cluster the dataset and obtain the initial prior anchor box parameters. However, the objects to be detected in the dataset generated in this paper significantly differ from those in the COCO dataset. Additionally, the K-means clustering algorithm requires manual setting of the initial points, and if the given initial points are unsuitable, it may result in a local optimal solution. To solve this problem, KCS-YOLO employs the K-means++ clustering algorithm to perform clustering on the labeled multi-dimensional target frames. The initial points are randomly selected from the entire dataset. This allows the algorithm to bypass local optimal solutions. After multiple iterations, a global optimal solution is obtained.

Through the K-means++ clustering algorithm, 12 anchor boxes suitable for traffic light detection are ultimately generated separately: (4,8), (8,14), (18,9), (24,8), (11,20), (23,12), (9,41), (14,27), (18,35), (12,57), (26,48), and (19,84). The clustering centers processed by the K-means++ clustering algorithm are shown in Figure 6.

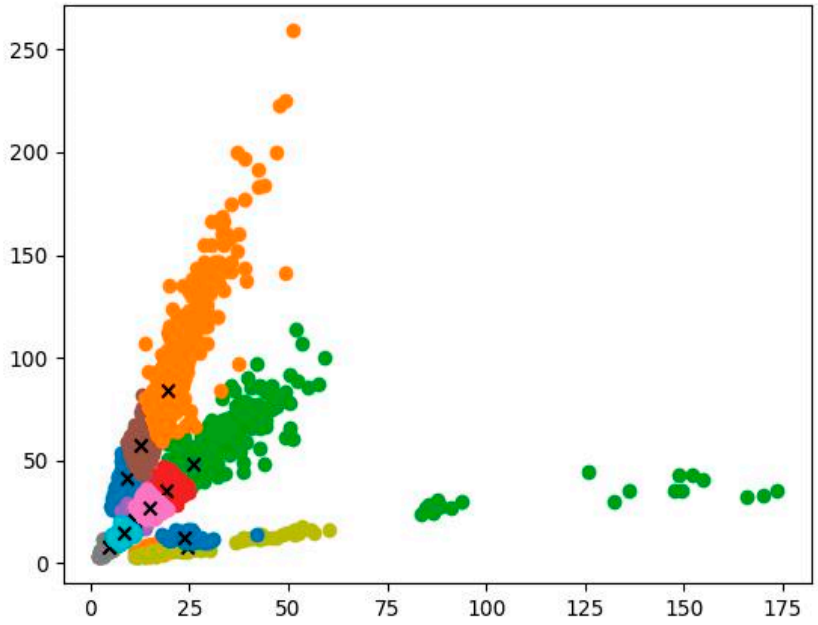

Figure 6. Distribution map of clustering centers. The dots with different colors represent anchor boxes of different sizes. There are twelve colors in the figure, denoting the twelve sizes of anchor boxes. The twelve "x" denote the cluster centers of the twelve anchor boxes, accordingly.

3.4.2. Incorporating an Attention Mechanism

Built upon the YOLOv5n algorithm, KCS-YOLO intends to improve the performance of the former by incorporating an attention mechanism. Among various attention mechanisms, such as coordinate attention (CA), CBAM, efficient multi-scale attention module (EMA), multidimensional collaborative attention (MCA), and similarity-based attention mechanism (SimAM), the optimal one is selected through a comparative experiment of these candidates. The chosen optimal attention mechanism is then incorporated into the traffic light detection algorithm to improve the perception of critical regions, thereby increasing the model's accuracy and achieving the best detection effect. The results derived from the comparative experiment are shown in Table 2.

Table 2. A comparison of performance measures of YOLOv5n algorithm with various attention mechanisms.

Attention Mechanisms	AP				mAP	F1				Params/M	FLOPs/G
	Red	Green	Yellow	Off		Red	Green	Yellow	Off		
YOLOv5n	0.9799	0.9588	0.9227	0.8922	0.9384	0.98	0.95	0.90	0.80	7.277	17.156
YOLOv5n_CA	0.9807	0.9889	1.0000	0.9500	0.9799	0.98	0.97	1.00	0.75	7.314	17.172
YOLOv5n_CBAM	0.9903	0.9719	1.0000	0.9712	0.9834	0.98	0.97	1.00	0.89	7.322	17.163
YOLOv5n_EMA	0.9917	0.9772	1.0000	0.9600	0.9822	0.99	0.96	1.00	0.88	7.331	18.129
YOLOv5n_MCA	0.9938	0.9765	1.0000	0.9500	0.9801	0.99	0.96	1.00	0.89	7.277	17.168
YOLOv5n_SimAM	0.9938	0.9765	1.0000	0.9500	0.9801	0.99	0.96	1.00	0.89	7.277	17.168

It is observed that the attention mechanism, i.e., CBAM, achieves better detection performance in mAP and F1 with a smaller algorithm parameter size. Moreover, CBAM achieves dual refinement of the input features by combining channel attention and spatial attention [5]. This enables it to focus on meaningful channels and spatial locations simultaneously, paying more attention to the core information features of traffic light images. Embedding the CBAM attention mechanism into the backbone of the YOLOv5n algorithm may significantly improve the algorithm's accuracy in detecting traffic lights in low visibility conditions. The structure of CBAM is shown in Figure 7. The CBAM attention mechanism is a lightweight module that sequentially calculates attention weights in the channel and spatial dimensions independently. It then multiplies them with the input features to adaptively adjust the input features, thereby enhancing the recognition capability of the features in low visibility conditions.

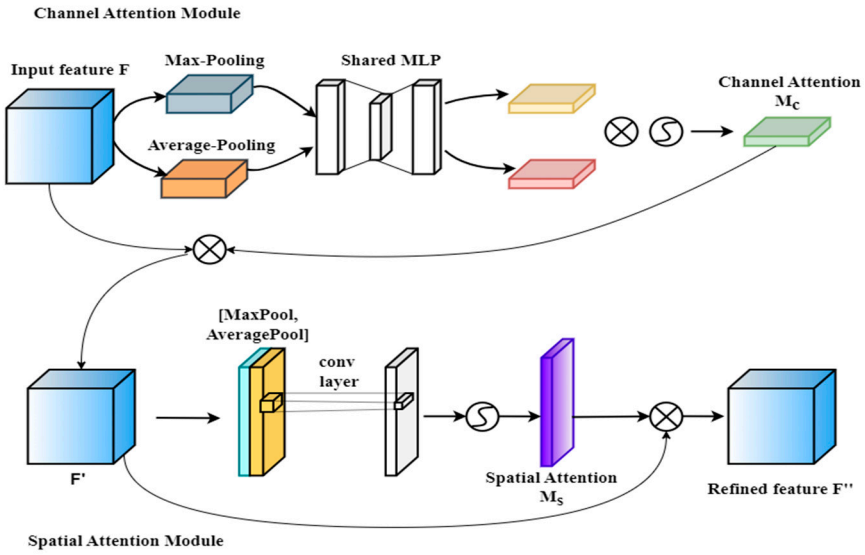

Figure 7. Structure of CBAM attention mechanism.

As shown in Figure 7, the channel attention module performs average pooling (AvgPool) and maximum pooling (MaxPool) on the input feature F, resulting in features F^c_{avg} and F^c_{max}. Then, these two metrics are fed into the multi-layer perceptron (MLP) for processing, and the weighting coefficient is calculated using the Sigmoid function (σ). Finally, the input feature F is multiplied by the weighting coefficient M_c to obtain a new feature F'. In the spatial attention module, the input feature F' undergoes pooling operations along the channel dimension, obtaining features F^s_{avg} and F^s_{max}. The resulting features are stacked. Subsequently, the convolution is used to adjust the channel number, calculating weighting coefficient M_s using the Sigmoid function (σ). Finally, the feature F' is multiplied by the weighting coefficient M_s to obtain a new feature F''.

3.4.3. A Small Object Detection Layer

Many traffic lights are spotted at a distant location, occupying a small size in the image. In low visibility scenarios, it becomes even more challenging for in-vehicle cameras to recognize distant traffic lights. To address this problem, KCS-YOLO incorporates a small object detection layer, represented by the box with red dashed lines shown in Figure 5. The original three-scale feature fusion in YOLOv5n is modified into a four-scale feature fusion. The 80×80 feature detection layer is upsampled by a factor of two and then fused with the newly added 160×160 feature detection layer; a prediction head for tiny object detection is introduced, enhancing the semantic information and feature representation of small targets.

4. Results and Discussion

To examine the performance of the proposed algorithm, i.e., KCS-YOLO, a benchmark is conducted experimentally by comparing this algorithm with Faster-RCNN, YOLOv5n, and its variants. Faster-RCNN is a state-of-the-art object detection technique, which is a two-stage deep learning object detector [21]. It identifies regions of interest first; these regions are then passed to a convolution neural network (CNN). Faster-RCNN combines two modules: (1) a deep convolution neural network for proposing regions; and (2) the Faster-RCNN detector that utilizes the proposed regions. Once again, for simplicity, we treat YOLOv5n and its variants as different algorithms. This section introduces the experiment set-up, ablation experiments, and comparison experiments.

4.1. Experiment Set-Up

The experiments are conducted using the operating system, Windows 11, with an NVIDIA GeForce RTX 4060 Laptop GPU, with 16 GB memory. Python 3.9 is employed as the programming language, and the training is based on the deep learning framework PyTorch 1.12.1. During the training process, we utilize the Adaptive Moment Estimation (Adam) optimizer, with a momentum factor setting of 0.937 and an initial learning rate of 0.001. A cosine annealing scheduler is employed to adjust the learning rate [22]. Label smoothing is applied to traffic light classification labels to prevent overfitting, with a smoothing value of 0.01.

4.2. Ablation Experiments

To evaluate the performance of KCS-YOLO, a series of ablation experiments are designed to compare it with YOLOv5n and its variants. While training, label smoothing is applied to the images to accelerate the convergence of KCS-YOLO and ensure the stability of the experimental sample data. The evaluation is performed every 10 epochs, and the experiment is conducted for a total of 300 epochs. Note that in machine learning, an epoch is defined as one complete pass through the entire training dataset; one pass in turn means a complete forward and backward pass through the entire training dataset. Table 3 provides the results derived from the ablation experiments. Given the results, the performance measures of KCS-YOLO can be directly compared against those of YOLOv5n and its variants.

In the case of YOLOv5n_F, no dehazing processing is made on the dataset. For YOLOv5n_C, the CBAM attention mechanism is embedded. In YOLOv5n_K, the K-means++ for re-clustering anchors is incorporated. In YOLOv5n_S, the small object detection layer is involved. In YOLOv5n_CK, both the CBAM attention mechanism and the K-means++ for re-clustering anchors are integrated. In YOLOv5n_CS, the CBAM attention mechanism and the small object detection layer are embedded. In YOLOv5n_KS, the K-means++ for re-clustering anchors and the small object detection layer are incorporated. Finally, KCS-YOLO incorporates the CBAM attention mechanism, the K-means++ for re-clustering anchors, and the small object detection layer.

Table 3. A comparison of the performance measures of KCS-YOLO against those of YOLOv5n and its variants.

Algorithms	AP				mAP	F1				Params/M	FLOPs/G
	Red	Green	Yellow	Off		Red	Green	Yellow	Off		
YOLOv5n_F	0.9708	0.9492	0.8804	0.7838	0.8960	0.94	0.91	0.85	0.84	7.277	17.156
YOLOv5n	0.9799	0.9588	0.9227	0.8922	0.9384	0.98	0.95	0.90	0.80	7.277	17.156
YOLOv5n_C	0.9903	0.9719	1.0000	0.9712	0.9834	0.98	0.97	1.00	0.89	7.322	17.163
YOLOv5n_K	0.9889	0.9703	0.9677	0.9652	0.9730	0.98	0.97	0.93	0.91	7.277	17.156
YOLOv5n_S	0.9913	0.9746	0.9706	0.9899	0.9816	0.98	0.97	0.95	0.90	7.436	20.734
YOLOv5n_CK	0.9863	0.9703	0.9843	0.9922	0.9832	0.98	0.97	0.96	0.97	7.322	17.163
YOLOv5n_CS	0.9905	0.9744	0.9563	0.9437	0.9662	0.98	0.96	0.92	0.83	7.481	20.741
YOLOv5n_KS	0.9863	0.9673	0.9743	0.9927	0.9800	0.98	0.97	0.95	0.97	7.436	20.734
KCS-YOLO	0.9917	0.9757	1.0000	0.9872	0.9887	0.98	0.97	1.00	0.90	7.481	20.741

Introducing the K-means++ algorithm, CBAM, and the small object detection layer leads to an improved performance in mAP and F1. Note that for comparison, YOLOv5n is treated as the baseline algorithm. Compared with YOLOv5n, KCS-YOLO outperforms in achieving: (1) the mAP of 98.87%, an increase of 5.03% from the baseline value of 93.84%; (2) the F1 scores on the traffic light dataset for red, green, yellow, and off with the values of 98, 97, 100, and 90%, increasing by 0, 2, 10, and 10% from their baseline values of 98, 95, 90, and 80%, respectively. Among all the algorithms listed in Table 3, KCS-YOLO shows the best overall performance in AP, mAP and F1, with comparable measures of Params and FLOPs.

To further compare the baseline algorithm, YOLOv5n, and the proposed one, KCS-YOLO, we examine vehicle traffic light detections under various conditions, i.e., night-time, distant targets, and foggy weather. Figure 8 shows the benchmark results.

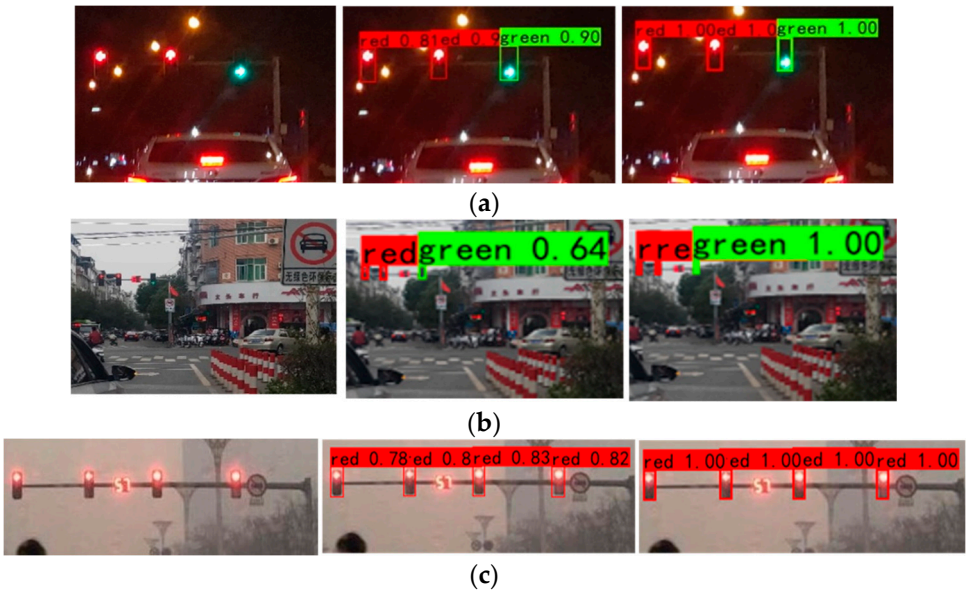

Figure 8. The experimental results of vehicle traffic light detections for the original image (**left**), the image detected by YOLOv5n (**middle**), and the image detected by KCS-YOLO (**right**): (**a**) night-time; (**b**) long-distance; (**c**) foggy conditions.

Under the night-time scenario, in the case of YOLOv5n, the detection accuracy for red, red, and green lights are 0.81, 0.9, and 0.9, respectively; while in the case of KCS-YOLO, the counterparts are 1.0, 1.0 and 1.0, accordingly. Note that the red, red, and green lights are located from the left to the right, as shown in Figure 8a. Under the long-distance scenario, compared with the baseline algorithm, KCS-YOLO reaches the green light detection

accuracy with the score of 1.0, an increase of 36% from its baseline value of 0.64. Under the foggy conditions, in the case of YOLOv5n, the detection accuracy for the red lights from the left to the right are 0.78, 0.8, 0.83 and 0.82, respectively, while in the case of KCS-YOLO, the detection accuracy for all red lights reaches the value of 1.0. The benchmark testing results indicate that KCS-YOLO outperforms YOLOv5n in traffic light detection accuracy in all three scenarios.

4.3. Comparison Experiments

To further examine KCS-YOLO performance on the traffic light dataset, a benchmark experiment is conducted to compare the proposed algorithm with YOLOv5n and Faster-RCNN. Figure 9a,b show the loss value curve and the mAP curve for each algorithm over the training and testing process, respectively.

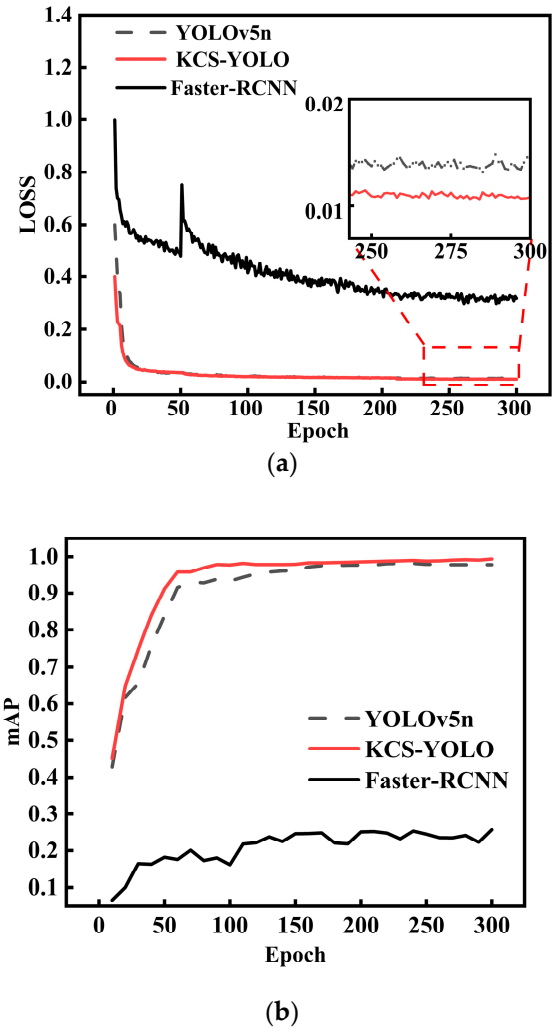

Figure 9. Results derived from benchmark experiments: (**a**) loss value curves; (**b**) mAP curves.

Figure 9a shows that within the first 50 epochs, the loss values for all the algorithms decrease rapidly. After training stabilized, KCS-YOLO outperforms YOLOv5n and Faster-RCNN in achieving a smaller loss value. From the mAP curves shown in Figure 9b, it is evident that for KCS-YOLO and YOLOv5n, the mAP values increase rapidly within the first 100 epochs. After training for 200 epochs, both algorithms tend to stabilize, and KCS-YOLO performs better than YOLOv5n in attaining a higher mAP value. In the case of Faster-RCNN, within the first 150 epochs, the mAP curve slowly increases, while over the range of 150 to 300 epochs, the mAP curve tends to be stabilized with small oscillations.

Table 4 lists the performance measures for all the three algorithms derived from the benchmark experiments. KCS-YOLO achieves the mAP value of 0.9887, an increase of 5.03% and 74.56% from the baseline value of 0.9384 (for YOLOv5n) and the corresponding value of 0.2431 for Faster-RCNN. The results shown in Table 4 achieve excellent agreement with those listed in Table 3. It is demonstrated that KCS-YOLO exhibits superior performance and achieves more precise traffic light detection and recognition. A close observation of Figure 9 and Table 4 discloses the following facts: (1) even though KCS-YOLO performs better than YOLOv5n in the loss value and mAP, the two algorithms are comparable in these performance measures; (2) Faster-RCNN performs much poorer than both KCS-YOLO and YOLOv5n in the metrics.

Table 4. The results of benchmark experiments.

Algorithms	AP				mAP	F1				Params/M	FLOPs/G
	Red	Green	Yellow	Off		Red	Green	Yellow	Off		
YOLOv5n	0.9799	0.9588	0.9227	0.8922	0.9384	0.98	0.95	0.90	0.80	7.277	17.156
Faster-RCNN	0.2877	0.2628	0.2213	0.2009	0.2431	0.38	0.37	0.22	0.24	41.327	251.437
KCS-YOLO	0.9917	0.9757	1.0000	0.9872	0.9887	0.98	0.97	1.00	0.90	7.481	20.741

The associated research findings reported in the literature may well explain the second observed fact. It is difficult for Faster-RCNN to detect small objects [23]. In traffic light detection, the targets are small, and may be occluded. This often leads to missed detections, thereby leading to the impaired performance of Faster-RCNN. In addition, Faster-RCNN suffers from incomplete information in feature extraction [24,25], especially under low visibility conditions, where it fails to adequately capture details, such as the specific features of traffic lights (color, shape, and size). Furthermore, Faster-RCNN has not fully leveraged data augmentation techniques, which prevents it from achieving high accuracy under low visibility conditions, resulting in weak model generalization and low mAP value.

5. Conclusions

In low visibility conditions, traffic light detection and recognition becomes more challenging. To improve the traffic light detection accuracy, this paper proposes and designs an algorithm, called KCS-YOLO, which is built upon YOLOv5n. KCS-YOLO is distinguished with the following features: (1) using the K-means++ clustering algorithm for the clustering of labeled multi-dimensional target frames; (2) embedding the CBAM attention mechanism and introducing a small-target detection layer to obtain more crucial traffic light features and enhance the detection capability; and (3) using the traffic light image dataset generated with the dark channel prior dehazing algorithm to eliminate foggy image noise, strengthen the detection and recognition capability of weak lighting at night-time, consequently boosting the information content of the images. To evaluate the performance of the proposed algorithm, we conduct numerous experiments.

Experimental results demonstrate that compared with YOLOv5n and its variants, KCS-YOLO exhibits the best overall performance in AP, mAP and F1 with the comparable measures of Params and FLOPs. In particular, KCS-YOLO achieves the mAP value of 0.9887, an increase of 5.03% from the respective indicator of 0.9384 for YOLOv5n. It is indicated that KCS-YOLO can enhance the detection and recognition capabilities of traffic lights in low visibility conditions. It is found that Faster-RCNN is not suitable for traffic light detection and recognition under low visibility conditions. The proposed algorithm does not pay special attention to other harsh conditions, e.g., sun glare, shadows, and highly reflective backgrounds. In the near future, the algorithm will be improved considering these harsh conditions. To verify the proposed algorithm, it is currently being tested on a low-speed mobile robot. The trade-off between the accuracy and computational efficiency of the algorithm will be explored. Once attaining acceptable computational efficiency, the proposed algorithm is to be examined on an autonomous road vehicle in real-time.

Author Contributions: Project administration, Q.Z.; Methodology, D.Z.; Data collection, H.L.; Writing—review and editing, Y.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the "Research Project of the Ministry of Housing and Urban-rural Development of the People's Republic of China, grant number 2022-K-079".

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the lead author.

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

References

1. Liu, D.; Mu, H.; He, Q.; Shi, J.; Wang, Y.; Wu, X. A Low Visibility Recognition Algorithm Based on Surveillance Video. *J. Appl. Meteor. Sci.* **2022**, *33*, 501–512. [CrossRef]
2. Viezee, W.; Evans, W.E. Automated Measurements of Atmospheric Visibility. *J. Appl. Meteorol.* **2013**, *22*, 1455–1461. [CrossRef]
3. Sabu, A.; Vishwanath, N. An improved visibility restoration of single haze images for security surveillance systems. In Proceedings of the 2016 Online International Conference on Green Engineering and Technologies (IC-GET), Coimbatore, India, 19 November 2016; pp. 1–5. [CrossRef]
4. Ait Ouadil, K.; Idbraim, S.; Bouhsine, T. Atmospheric visibility estimation: A review of deep learning approach. *Multimed Tools Appl.* **2024**, *83*, 36261–36286. [CrossRef]
5. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. CBAM: Convolutional Block Attention Module. In *Computer Vision—ECCV 2018*; Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y., Eds.; Lecture Notes in Computer Science; Spring: Cham, Switzerland, 2018; Volume 11211. [CrossRef]
6. Tian, E.; Kim, J. Improved Vehicle Detection Using Weather Classification and Faster R-CNN with Dark Channel Prior. *Electronics* **2023**, *12*, 3022. [CrossRef]
7. Yang, L. A Deep Learning Method for Traffic Light Status Recognition. *J. Intell. Connect. Veh.* **2023**, *6*, 173–182. [CrossRef]
8. Possatti, L.C.; Guidolini, R.; Cardoso, V.B.; Berriel, R.F.; Paixão, T.M.; Badue, C.; De Souza, A.F.; Oliveira-Santos, T. Traffic Light Recognition Using Deep Learning and Prior Maps for Autonomous Cars. In Proceedings of the 2019 International Joint Conference on Neural Networks (IJCNN), Budapest, Hungary, 14–19 July 2019; pp. 1–8. [CrossRef]
9. Du, L.; Chen, W.; Fu, S.; Kong, H.; Li, C.; Pei, Z. Real-time Detection of Vehicle and Traffic Light for Intelligent and Connected Vehicles Based on YOLOv3 Network. In Proceedings of the 2019 5th International Conference on Transportation Information and Safety (ICTIS), Liverpool, UK, 14–17 July 2019; pp. 388–392. [CrossRef]
10. Kou, A. Detection and recognition of traffic signs based on improved deep learning. *Int. Core J. Eng.* **2020**, *6*, 208–213. [CrossRef]
11. Mondal, K.; Rabidas, R.; Dasgupta, R. Single image haze removal using contrast limited adaptive histogram equalization based multiscale fusion technique. *Multimed. Tools Appl.* **2024**, *83*, 15413–15438. [CrossRef]
12. Li, Z.; Zhang, W.; Yang, X. An Enhanced Deep Learning Model for Obstacle and Traffic Light Detection Based on YOLOv5. *Electronics* **2023**, *12*, 2228. [CrossRef]
13. Mao, K.; Jin, R.; Ying, L.; Yao, X.; Dai, G.; Fang, K. SC-YOLO: Provide Application-Level Recognition and Perception Capabilities for Smart City Industrial Cyber-Physical System. *IEEE Syst. J.* **2023**, *17*, 5118–5129. [CrossRef]
14. Appiah, E.O.; Mensah, S. Object detection in adverse weather condition for autonomous vehicles. *Multimed. Tools Appl.* **2024**, *83*, 28235–28261. [CrossRef]
15. Li, Z.; Zeng, Q.; Liu, Y.; Liu, J.; Li, L. An improved traffic lights recognition algorithm for autonomous driving in complex scenarios. *Int. J. Distrib. Sens. Netw.* **2021**, 1–17. [CrossRef]
16. HumanSignal/labelImg. Available online: https://github.com/HumanSignal/labelImg (accessed on 26 June 2024).
17. He, K.; Sun, J.; Tang, X. Single image haze removal using dark channel prior. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 1956–1963. [CrossRef]
18. Peng, Y.-T.; Cao, K.; Cosman, P.C. Generalization of the Dark Channel Prior for Single Image Restoration. *IEEE Trans. Image Process.* **2018**, *27*, 2856–2868. [CrossRef] [PubMed]
19. 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]
20. Fan, J.; Huo, T.; Li, X. A Review of One-Stage Detection Algorithms in Autonomous Driving. In Proceedings of the 2020 4th CAA International Conference on Vehicular Control and Intelligence (CVCI), Hangzhou, China, 18–20 December 2020; pp. 210–214. [CrossRef]
21. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [CrossRef] [PubMed]
22. Dou, Z.; Zhou, H.; Liu, Z. An Improved YOLOv5s Fire Detection Model. *Fire Technol.* **2024**, *60*, 135–166. [CrossRef]
23. Tang, L.; Li, F.; Lan, R.; Luo, X. A small object detection algorithm based on improved faster RCNN. In Proceedings of the International Symposium on Artificial Intelligence and Robotics, Fukuoka, Japan, 21–22 August 2021. [CrossRef]

24. Cao, C.; Wang, B.; Zhang, W.; Zeng, X.; Yan, X.; Feng, Z.; Liu, Y.; Wu, Z. An improved Faster R-CNN for small object detection. *IEEE Access* **2019**, *7*, 106838–106846. [CrossRef]
25. Ji, H.; Gao, Z.; Mei, T.; Li, Y. Improved Faster R-CNN with multiscale feature fusion and homography augmentation for vehicle detection in remote sensing images. *IEEE Geosci. Remote Sens. Lett.* **2019**, *16*, 1761–1765. [CrossRef]