electronics

Article

Traffic Light Detection by Integrating Feature Fusion and Attention Mechanism

Chi-Hung Chuang [1], Chun-Chieh Lee [2], Jung-Hua Lo [3,*] and Kuo-Chin Fan [2]

[1] Department of Information and Computer Engineering, Chung Yuan Christian University, Taoyuan 320314, Taiwan; chchuang@cycu.edu.tw
[2] Department of Computer Science and Information Engineering, National Central University, Taoyuan 320317, Taiwan; jackcclee@cc.ncu.edu.tw (C.-C.L.); kcfan@csie.ncu.edu.tw (K.-C.F.)
[3] Department of Applied Informatics, Fo-Guang University, Yilan County 26247, Taiwan
* Correspondence: jhlo@mail.fgu.edu.tw

Abstract: Path planning is a key problem in the design of autonomous driving systems, and accurate traffic light detection is very important for robust routing. In this paper, we devise an object detection model, which mainly focuses on traffic light classification at a distance. In the past, most techniques employed in this field were dominated by high-intensity convolutional neural networks (CNN), and many advances have been achieved. However, the size of traffic lights may be small, and how to detect them accurately still deserves further study. In the object detection domain, the scheme of feature fusion and transformer-based methods have obtained good performance, showing their excellent feature extraction capability. Given this, we propose an object detection model combining both the pyramidal feature fusion and self-attention mechanism. Specifically, we use the backbone of the mainstream one-stage object detection model consisting of a parallel residual bi-fusion (PRB) feature pyramid network and attention modules, coupling with architectural tuning and optimizer selection. Our network architecture and module design aim to effectively derive useful features aimed at detecting small objects. Experimental results reveal that the proposed method exhibits noticeable improvement in many performance indicators: precision, recall, F1 score, and mAP, compared to the vanilla models. In consequence, the proposed method obtains good results in traffic light detection.

Keywords: object detection; attention mechanism; feature pyramid; self-driving car; traffic light

Citation: Chuang, C.-H.; Lee, C.-C.; Lo, J.-H.; Fan, K.-C. Traffic Light Detection by Integrating Feature Fusion and Attention Mechanism. *Electronics* **2023**, *12*, 3727. https://doi.org/10.3390/electronics12173727

Academic Editor: Donghyeon Cho

Received: 7 August 2023
Revised: 28 August 2023
Accepted: 30 August 2023
Published: 4 September 2023

1. Introduction

The current trend of autonomous driving cars is gaining momentum, and autopilot vehicles are approaching a point where they can be seen on the roads. These AI-powered vehicles are intelligent enough to mimic human perception and make finetuned movements; however, there are still many issues that need to be resolved.

According to the Society of Automotive Engineers International (SAE International) [1], autonomous driving is classified into six levels: Levels 0 to 2 are considered driver assistance systems, where the human driver is still in control, but the vehicle has features such as automatic emergency braking and lane departure correction. Levels 3 to 5 are categorized as automated driving systems, where the vehicle takes control, and when all conditions are met, the automated driving function can be activated, and the vehicle can operate on the road.

This study is inspired by the environment and scene detection capabilities required for autonomous driving from Level 3 to Level 4. In the development of autonomous driving technology, the accurate recognition of traffic signals is crucial for achieving autonomous driving, especially in traffic violation detection, which brings many benefits. Through object detection technology, traffic violations such as running red lights, illegal parking, speeding, failure to yield to pedestrians, and wrong-way driving can be detected and

identified, allowing for real-time alerts and warnings. This helps to reduce the occurrence of traffic accidents and ensure road safety.

Traffic light detection is important for automatic driving support systems. In [2,3], two-stage approaches were proposed, where the traffic light area was first segmented or detected, and then the detected region was further used for recognition. The concept of salient traffic lights was introduced in [4] and a related dataset with annotated salience properties was constructed and used for model training. These works either applied two-stage processing or required new annotated datasets. In this study, we aim to develop the combination of a parallel residual bi-fusion feature pyramid network and a self-attention mechanism for one-stage traffic light recognition. The goal is to improve the vehicle's recognition and decision-making abilities regarding traffic signals, as well as learn the truly important features and their correlations within the image. However, the size of traffic lights may be small, and how to precisely detect them is still worthy of in-depth study. The main contribution of this paper is showing that incorporating either parallel residual bi-fusion feature pyramid network (PRB-FPN) or attention mechanisms (coordinate attention, CA or transformer block, T) into the vanilla models will greatly improve their performance in small object detection. Also, as there are many abbreviations throughout this paper, we include a description for each abbreviation at the end of this article.

Through this approach, it will be possible to enhance the safety and efficiency of autonomous vehicles in complex road environments and provide practical application value in addressing issues such as running red lights and failure to yield to pedestrians in traffic violation detection. It is expected that these research directions will contribute to the advancement and application of autonomous driving technology, ultimately contributing to the construction of a safer and more efficient transportation environment.

2. Related Work

In recent years, research in the field of object detection has garnered widespread attention, with extensive applications in the field of autonomous driving. To achieve high precision and efficiency in object detection, feature pyramids have been widely used as an effective method. A feature pyramid processes feature maps at different scales to extract feature information at various scales, enabling the detection of objects of different sizes. In the context of autonomous driving applications, objects often appear at different scales and orientations, making feature pyramids crucial for achieving comprehensive object detection. However, traditional feature pyramids suffer from issues such as loss of spatial information and high computational costs.

In recent years, some new object detection methods have replaced feature pyramids with the use of self-attention mechanisms to more effectively capture multiscale features of objects and improve detection accuracy. Self-attention mechanisms allow the model to focus on relevant regions in an image, enhancing its ability to recognize objects at different scales and orientations. This approach helps address the limitations of traditional feature pyramids and contributes to more accurate and efficient object detection in the field of autonomous driving.

2.1. Object Detection

Currently, deep learning-based object detection methods mainly include one-stage and two-stage approaches. One-stage methods, such as YOLOv4 [5], YOLOv7 [6], and SSD [7], directly predict object locations and categories together. They have advantages such as fast inference speed and accurate localization. Two-stage object detection methods, such as faster R-CNN [8] and mask R-CNN [9], first extract object proposals and then classify them. They typically achieve better detection accuracy and stronger generalization but are relatively slower in terms of speed. In addition, there are other object detection methods, such as those based on attention mechanisms, pose-based methods, and a combination of deep learning and traditional approaches. Each method has its strengths and weaknesses, and the choice and optimization should be made based on specific application

scenarios and requirements. You only look once [10], also known as the first version of the YOLO family, which has the advantage of fast processing speed, enabling real-time object detection. However, it exhibits lower accuracy in detecting small objects and dense object regions. YOLOv2 (YOLO9000) [11], addressed the limitations of [10] and improved its accuracy while maintaining faster processing speed. However, it still faces challenges in accurately detecting small objects. YOLOv3 [12] introduced a new backbone architecture called Darknet-53, and it also incorporated the FPN to capture features at different scales. The advantage of [12] is that it provides better accuracy while maintaining relatively faster processing speed. However, it still faces limitations in accurately detecting small objects and dense scenes. In [5], the authors incorporate CSPDarknet53, SPP module, SAM, and PANet, achieving significant improvements in both accuracy and speed. It introduces advanced features such as improvements in detecting small objects. However, it requires higher computational resources and longer training time. YOLOv5 [13] has a similar architecture to YOLOv4 but introduces the Focus architecture and replaces the SPP module with the SPPF module. The advantages of it are its concise code, faster training speed, and ease of integration with other architectures. However, it may not perform as well as YOLOv4 in certain use cases, as indicated in some reports in [5]. YOLOv6 [14] introduces RepVGG [15] and replaces the original Backbone architecture with EfficientRep [16]. The neck is composed of the Rep-PAN architecture. The advantages of [14] include further improvements in detection accuracy and performance. In [6], they introduce a new backbone architecture called E-ELAN and also improve the reparameterization module strategy, known as Repconv, which applies to various architectures. These advancements have led to enhanced accuracy and speed in object detection. YOLOv8 [17] is an anchor-free detection model, which reduces the number of predicted boxes, leading to faster NMS speed. How to combine newer technologies to better deal with some existing problems in small traffic light images and to further improve performance are the main research directions at present.

2.2. Feature Pyramid

FPN [18] ensures that each feature map in the pyramid retains strong features. It is built upon the foundation of SSD by adding a top-down pathway to integrate features from different scales. Through this pathway, the lower-level features are enriched with better semantic information, playing a crucial role in the recognition of small objects. PANet [19] builds upon FPN by adding a bottom-up path augmentation. In the lower layers, there are many crucial features, but as the information is passed upwards through multiple layers, there is a relatively significant loss of information. To address this issue, PANet introduces a bottom-up pathway, allowing information to flow from lower to higher layers, mitigating the problem of information loss and enhancing the overall feature representation. PRB-FPN [20] introduces a CORE bi-fusion module, where the CORE module combines features and feature position information from the current layer, the previous layer, and the next layer. This bi-directional feature fusion approach helps to recover lost information by leveraging the features from lower layers, thereby improving the preservation of positional information that may be lost or displaced due to pooling effects. After the first step of feature fusion using the CORE module, the resulting feature information undergoes multiple convolutions through the Convolutions module. The output from this process is then passed to the next layer's CORE module as feature information from the previous layer. As this process of feature fusion with CORE and Convolutions modules continues multiple times from top to bottom, it forms a richer context of semantic features, leading to a significant improvement in the accuracy of features for detecting small objects.

Another module called Re-CORE is utilized, where the CORE module from the upper layer directly connects to the CORE module in the lower layer. In this Re-CORE module, the feature information fused by the CORE module in the upper layer is bypassed through the Convolutions module, retaining a portion of the features. These retained features are then added to the feature fusion process in the CORE module of the lower layer. This residual fusion approach allows the new feature pyramid to be easily trained, compatible

with different Backbone networks, and retaining a portion of the original features to further improve the accuracy of small object features. In the end, to enhance the accuracy of both small and large objects, the previous modules focus on top-down fusion, while the final module in the bi-fusion structure, the BFM (bottom-up feature module), integrates features information from different scales in a bottom-up manner. This comprehensive approach aims to improve the accuracy of both small and large object features.

2.3. Attention Mechanism

Self-Attention is a mechanism based on the Transformer [21]. In recent years, Self-Attention has been widely applied in fields such as object detection and segmentation. Models like DETR [22] and VoVNet [23] have utilized the Self-Attention mechanism and achieved impressive performance. ViT [24] is a visual attention network architecture derived from the model [21], designed for image visual tasks, and it introduces the concept of self-attention, offering a novel approach. In object detection, Self-Attention can help the model capture regions of interest for objects, thereby enhancing the accuracy of object detection. This method treats each position in the feature map as an object and utilizes mechanisms like QKV (Query-Key-Value) to weigh the interactions between these objects. When using Self-Attention, an Attention Matrix is often employed to measure the similarity between different positions, and this matrix is then applied to the feature map to weigh the important feature locations, resulting in more discriminative feature representations. The key idea of [24] is to divide the image into several patches and transform these patches into a sequential format for input. Each patch undergoes linear mapping and positional embedding before being passed into the Transformer Encoder. The self-attention mechanism in the model can search for the correlations between the entire image and the patches, as well as among the patches themselves, thus achieving feature extraction and focusing on crucial features. Its strengths lie in overcoming the limitations of traditional CNNs, especially when dealing with large-sized images. However, it also faces challenges, such as computational efficiency and the demand for a large amount of training data.

To deal with the above issues, numerous researchers have been continuously working on improving the architecture and training strategies to expand its application scope and enhance its performance. In [25], the authors analyze the incorporation of Self-Attention into one-stage object detection models using different architectures and conclude that using skip connections can yield better results. In the past, channel attention mechanisms like SENet [26] and CBAM [27] often tended to overlook the spatial information of objects. Therefore, the method called coordinate attention (CA) introduced in [28] aims to enhance the model's ability to localize objects and extract regional features. It decomposes the channel attention mechanism into two feature maps. One of them captures feature correlations along the spatial direction, while the other retains the position information of the objects. This approach strengthens the focus on target objects and improves the model's ability to localize them, achieving the desired effect of emphasizing important features without significantly increasing computational complexity.

2.4. Traffic Light Detection

Path planning is a key issue in the design of autonomous driving systems, and accurate traffic light detection is crucial for autonomous vehicle pathfinding. In the work of [2], it presents a two-staged deep learning-based traffic light recognition method that consists of candidate detection and classification stages. For candidate detection, it employs a binary semantic segmentation network that is suitable for detecting small objects such as traffic lights. The traffic lights classification stage classifies the types of traffic lights by using an input image corresponding to a candidate region obtained from the candidate detection stage. Another two-stage approach was proposed in [3]. In this paper, they first apply the trained YOLOv5s to detect the position of the traffic light in the target scene, then extract the target area and perform image processing, judging whether the traffic light is vertical or horizontal, and then cut the picture into small units to avoiding image distortion.

Gaussian noise removal is used for preprocessing, and then the processed image is input to the AlexNet network for recognition and obtaining the final decision. In [4], the authors propose a model that focuses on the task of traffic light detection. First, the paper shows that salient traffic lights can significantly influence a driver's future decisions. Next, it uses this salience property to construct the LAVA Salient Lights Dataset and the dataset is the first US traffic light dataset with an annotated salience property. Finally, the paper uses this data set to train a deformed DETR (DEtection TRansformer) object detection model and shows that it performs better.

3. Methodology

The feature extraction backbone in this study utilizes CSPDarknet53 and the Backbone from [6], which we refer to as E-ELAN. With this architecture, we aim to enhance the model's recognition capability of small objects, so we incorporate the methods in [20] to retain more small object features. However, we observed that due to the dominant role of high-quality, high-intensity convolutions in the model, important resources were dispersed to less critical objects. This finding led us to a new requirement of finding a method that enables the model to learn feature correlations and allocate attention resources to truly important objects. Transformer block and coordinate attention were found to meet these requirements. By combining the aforementioned requirements and choices, we arrived at the overall architecture of the proposed model shown in Figure 1.

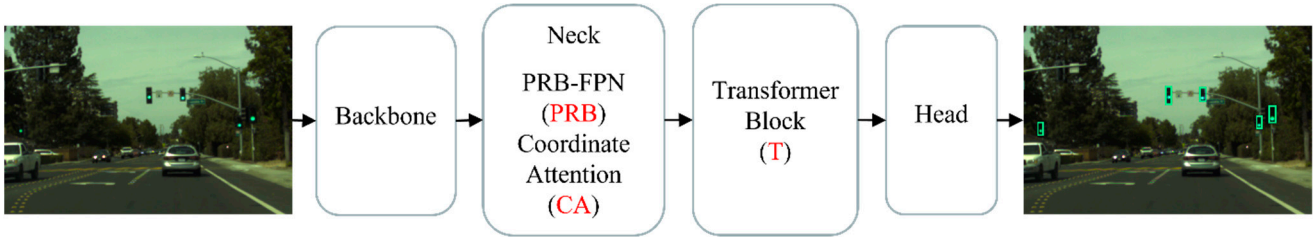

Figure 1. The diagram of the proposed model.

The current mainstream approaches for incorporating the transformer block into image tasks fall into two categories. The first approach involves adding it to the Backbone to contribute to feature extraction, while the second approach integrates it into the Neck to enhance feature fusion. We have opted for the second approach. The decision to place the transformer block after PRB rather than in the Backbone or before PRB is based on the specific requirements we discussed earlier. Our goal is to fully leverage the capabilities and characteristics of feature extraction, feature fusion, and attention mechanisms. PRB is a highly efficient multiscale object feature fusion method, and it better preserves object features in earlier stages of processing. On the other hand, the transformer block consists of multiple self-attention mechanisms that consider the global context of the entire image, leading to significant computational complexity and resource consumption. By placing the transformer block after PRB, we can reduce the overall computational burden. We first extract and retain detailed feature maps through PRB-FPN and then apply the transformer block, allowing each method to maximize its benefits without redundant computations. In our research, we introduced the coordinate attention (CA) scheme to the model to focus on the correlations between feature channels. Therefore, we chose to place it in the Neck region during model implementation.

3.1. PRB-FPN

We opted to use the popular one-stage object detection method along with the improved and refined PRB, which is based on FPN. The feature fusion approach of PRB has shown remarkable effectiveness in detecting small objects. By fusing feature maps from different scales, the model gains the ability to handle object detection of various sizes.

In the Backbone, the input image undergoes feature extraction, resulting in the generation of a feature map pyramid with five different scales. The bi-fusion module follows a method where three layers are extracted and fused from the feature pyramid, which is then passed to the next layer as information from the upper layer. This process is repeated to retain small object features from the top layer. When combined with the Backbone, the model in this research contains three such bi-fusion modules as shown in Figure 2a.

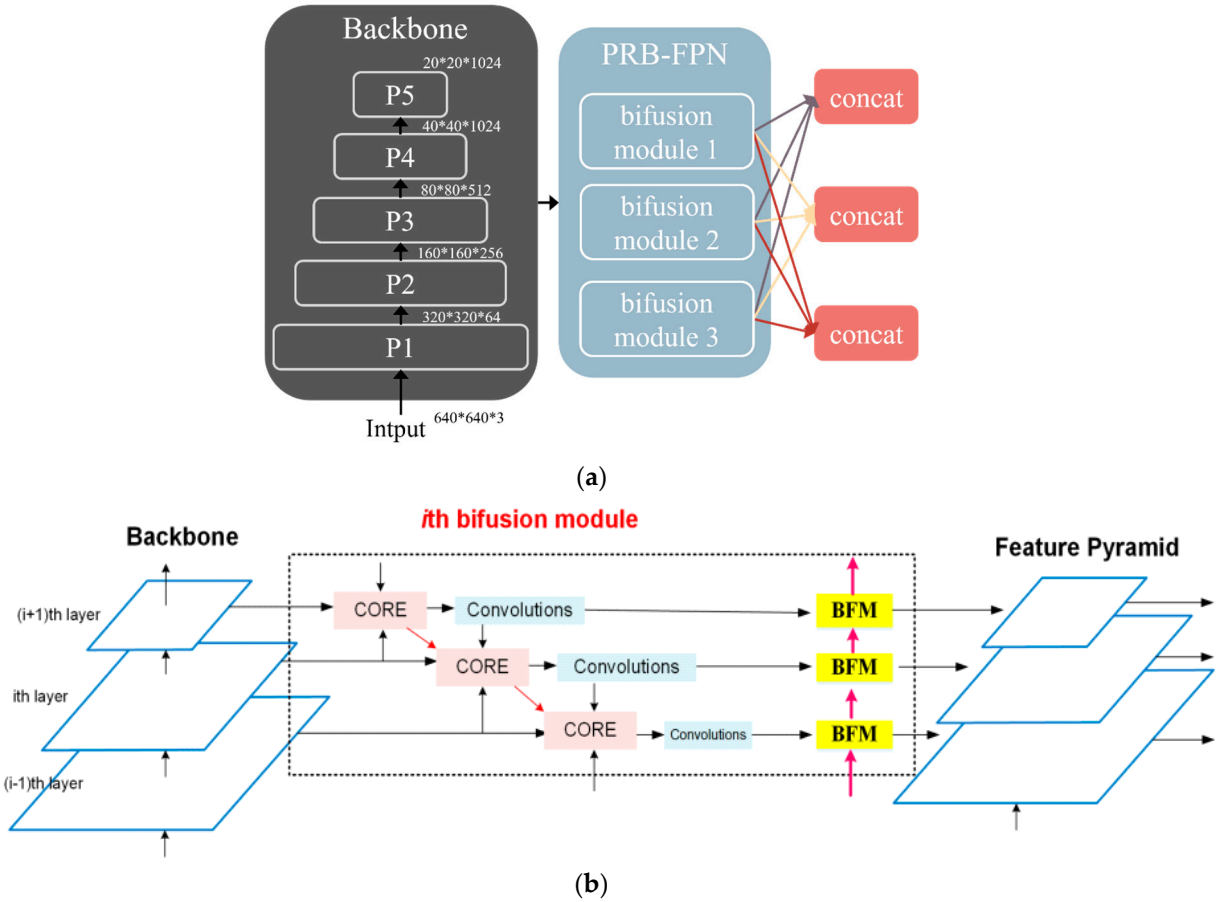

Figure 2. (**a**) Feature extraction and feature fusion architecture. The bi-fusion modules are sequentially named based on the top-down feature information propagation. Note that the symbol * represents the multiplication operation. (**b**) Bi-fusion module architecture for fusing adjacent layer contextual features [20].

In Figure 2a, in bi-fusion module 1, the feature fusion is performed on the feature maps P5, P4, and P3 extracted from the Backbone. In bi-fusion module 2, the feature fusion is performed on the feature maps P4, P3, and P2 from the Backbone. Lastly, in bi-fusion module 3, the feature fusion is performed on the feature maps P3, P2, and P1 from the Backbone. After the feature fusion in the three bi-fusion modules, each module has feature maps with channel sizes of 128, 256, and 512, respectively. We extract feature maps with the same channel size from the three modules and concatenate them together. At this stage, the information transfer task from the input to the Backbone and then to PRB is completed.

In Figure 2b, the CORE module combines the feature maps and feature location information of the current layer, the upper layer, and the lower layer. This top-down and bottom-up two-way feature fusion method restores the lost information using the lower-level features and maintains the position information that FPN is easily lost or displaced due to the pooling operations.

The features fused in the first step of the CORE module are convolved multiple times and then used as the fused features of the next layer. The above process is performed

multiple times from top to bottom. The feature fusion operation can obtain more contextual semantic features, which can significantly improve the accuracy of small object detection. Finally, to improve the detection accuracy of both small and large objects, the last BFM block in the bi-fusion module integrates features of different scales from bottom to top, while previous operations focus on top-down.

3.2. Coordinate Attention

When incorporating the coordinate attention into the model, we choose to place it after the SPPCSPC module in the neck region. This is because this layer performs max-pooling with three different kernel sizes and then concatenates and organizes these feature maps. Therefore, by adding the coordinate attention layer after this process, we can focus on the most relevant and useful features in the model.

3.3. Transformer Block

The transformer block, also known as the transformer encoder, was originally developed for natural language processing tasks, including both encoder and decoder components. The encoder part of the transformer is used to determine the sequential relationship of a given text, which we adapt for image tasks to analyze the relationships between object features. This allows the model to understand the correlations between features and between the entire image and features, learning the significance of important and unimportant features, and mimicking the human visual mechanism when observing objects, where less or no attention is given to background elements.

In our model, the transformer block differs from the encoder architecture used in [24]. We remove the embedding of positional information and the activation function from the layer normalization and MLP modules, replacing them with two fully connected layers, simplifying the internal structure of the transformer block, as shown in Figure 3.

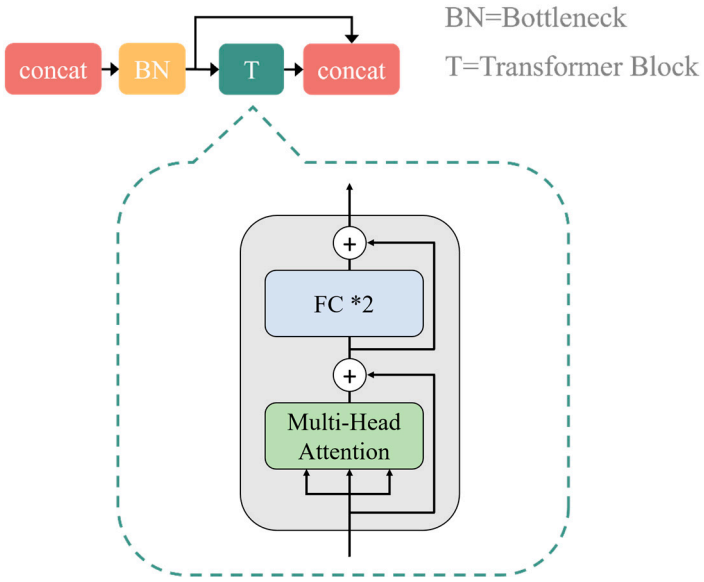

Figure 3. The transformer block architecture, note that the symbol * represents module duplication.

When the feature maps enter the transformer block, we retain a portion of the original feature maps and concatenate them with the feature maps processed through the multi-head self-attention mechanism. The QKV in this multi-head self-attention mechanism is obtained using the Linear method, eliminating the need for extensive computational resources. Subsequently, we concatenate a portion of the feature maps that have not been processed using MLP with the feature maps that have been processed, and output the combined features, completing all tasks for this stage of feature map processing.

3.4. Strategy and Implementation

Starting from the feature maps processed using PRB, the architecture is shown in Figure 4. The three bi-fusion modules, each with feature maps of the same channel size, are concatenated together. These concatenated feature maps then pass through the bottleneck module, which consists of a 1×1 convolution and a 3×3 convolution. The purpose of this module is to efficiently change the channel size of the feature maps and combine the concatenated information. Next, a skip connection is added to connect the feature maps that have undergone the transformer block with the feature maps before the self-attention mechanism. This strategy is introduced to preserve a portion of the feature information that has not yet been processed by the self-attention mechanism, which is meaningful and beneficial for improving the model's performance. After the skip connection, another bottleneck module with 1×1 and 3×3 convolutions is applied, followed by a 3×3 convolution in the Conv module. In the PRB output, we select three different layers with varying channel sizes as the final output: $20 \times 20 \times 1024$, $40 \times 40 \times 512$, and $80 \times 80 \times 256$.

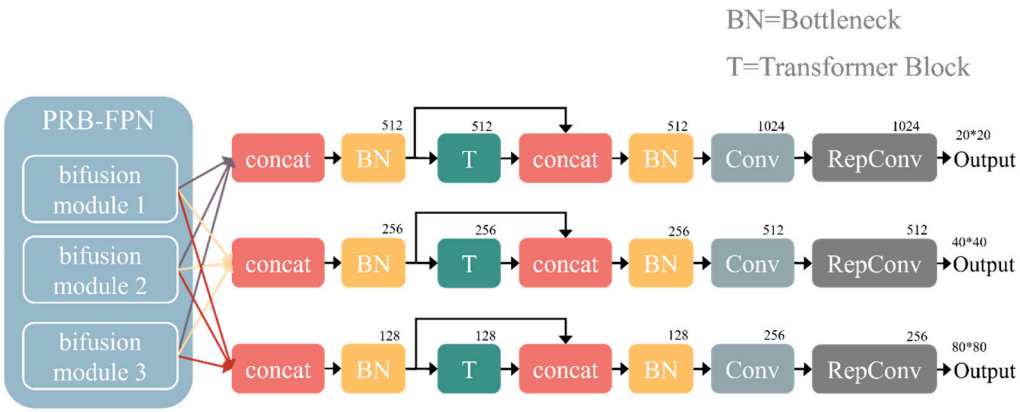

Figure 4. The strategy and implementation diagram, note that the symbol * represents the multiplication operation.

In the Head, we retained some of the techniques used in YOLOv7 Head, including the same convolutional layers and RepConv layers. The reason behind this decision is that such an approach effectively maintains the model's efficiency while preserving high accuracy and performance characteristics.

4. Experiments and Results Analysis

The experimental environment for this paper is Ubuntu 18.04.5 LTS, Intel® Core™ i7-11700K @ 3.60 GHz CPU, NVIDIA Geforce RTX 3090 GPU, 64 GB CPU Memory, 24 GB GPU Memory, Python 3.8.8, torch 1.11.0 + cu113. The optimizer of the neural network model is Adam [29], with a learning rate of 0.0001. The input size of the images is 640×640, and the batch size is 2. A total of 300 epochs of iterative training was conducted.

This paper employs the Adam optimizer, while the original [20] used the SGD [30]. The choice of using Adam is that transformer-related models typically have a large number of parameters with different scales. Using the Adam optimizer allows for better adjustment of learning rates and helps to avoid issues such as gradient explosions, thus leading to more effective training of the model.

4.1. Bosch Small Traffic Lights Dataset

The dataset used in this paper is the Bosch Small Traffic Lights Dataset [31], which was released in the year 2017 by the Bosch North America Research Institute located in Palo Alto, California, in the San Francisco Bay Area. It is designed for deep learning models related to traffic light tasks, including traffic light detection, tracking, and classification. This dataset primarily focuses on collecting and annotating small-sized traffic lights. There

are differences in annotation and processing between the training and testing datasets. Therefore, we describe each dataset separately.

4.1.1. The Training Dataset

The images were captured by mounting a camera on a moving car, which was driven along Royal Avenue in the San Francisco Bay Area, California. The Bosch dataset is a significant contribution to achieving fully autonomous driving technology in urban environments. To assess the capabilities of models trained on this dataset, it covers a wide range of real-world road scenes and weather conditions, including busy city streets; suburban multilane roads; construction zones; various weather conditions such as sunny, overcast, and rainy; as well as blurred traffic lights, among others. The images were captured using a red-clear-clear-blue filter, and the camera resolution is 1280 × 720 pixels, as shown in Figure 5.

Figure 5. The Bosch small traffic lights dataset.

The training set consists of a total of 5093 images, and the annotations are recorded at a frequency of approximately 2 s per annotation. There is a total of 10,756 annotated traffic lights, with 170 of them partially occluded. The most common types of traffic lights in the dataset are "green," "yellow," and "red." Due to the difference in data collection frequency and the update frequency of each traffic light, many traffic lights may appear as "off" rather than one of the three common types mentioned above. In such cases, the traffic lights are annotated as the "off" type instead of the actual light type at that moment. This is necessary for single-frame classification tasks, as the classification is based on the current image, and the "off" type is essential for an accurate representation of the data.

In the training set, the widths of different traffic lights vary from approximately 1 to 98 pixels, with an average width of 11.1 pixels. The dataset even includes traffic lights with a width of only 1 pixel, as shown in Table 1. This dataset presents a challenging scenario for evaluating the model's ability to recognize small objects.

Table 1. The sizes of traffic lights in the training set (unit: pixels).

	Minimum	Average	Median	Maximum
Width	1.12	11.18	8.55	98.0
Height	0.25	24.32	18.93	207.0
Area	0.28	404.52	158.8	20,286.0

4.1.2. The Test Dataset

The test set was collected by placing a camera on a moving car to capture images while driving along University Avenue in Palo Alto, California, in the San Francisco Bay Area. The environment, image filter, and camera resolution used for the test set are the

same as those used in the training set. The test set consists of 8334 consecutive images, with annotations provided at a frequency of approximately 15 frames per second (FPS). It includes a total of 13,486 annotated traffic lights, out of which 2088 are partially occluded. The test set's annotation types are not identical to the training set, but they still include the categories of "green", "yellow", "red", and "off", as seen in the training set.

In the test set, the width of different traffic light sizes varies approximately from 1 to 48 pixels, with an average of 9.4 pixels. Similar to the training set, the test set also includes traffic lights with a width of only 1 pixel, as shown in Table 2. These characteristics of the dataset pose a challenge to the model's ability to identify small objects accurately.

Table 2. The sizes of traffic lights in the test set (unit: pixels).

	Minimum	**Average**	**Median**	**Maximum**
Width	1.875	9.430	8.500	48.375
Height	3.250	26.745	24.500	104.500
Area	11.718	313.349	212.109	4734.000

4.2. Evaluation Metrics

Precision (P) is a metric for measuring the accuracy of model predictions. The calculation formula is as follows:

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

Recall (R) is a metric for measuring the detection capability of a model. The calculation formula is as follows:

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

In the above two formulas, TP denotes true positive, FP denotes false positive, and FN denotes false negative.

The $F1$ *score* is the weighted average of precision and recall and is used to comprehensively evaluate the accuracy and detection capability of a model. The calculation formula is as follows:

$$F1\ Score = \frac{2 \times P \times R}{P + R} \tag{3}$$

Average precision (AP) and mean average precision (mAP) are performance metrics usually used to evaluate object detection models, and these two evaluation indicators are also calculated in our experiments.

$$AP = \int_0^1 P(R) dR \tag{4}$$

The calculation of AP is associated with a certain object class, where $P(R)$ represents the precision P with the recall R given some decision threshold.

$$mAP = \frac{\sum_i AP_i}{C} \tag{5}$$

In the calculation of mAP, AP_i indicates the AP of class i and C is the number of classes.

4.3. Experimental Results

Among various object detection models, YOLO has the advantage of good detection performance and fast inference speed, making it a representative one-stage object detection model. Therefore, this research uses the YOLO series of object detection models as the basis, combined with PRB, attention mechanisms for training and testing on the traffic light dataset, and compares the results with the proposed model architecture in this study.

Firstly, we conduct experiments to compare the performance of our proposed model with other object detection models, to validate the detection capability of our proposed model architecture, as shown in Table 3.

Table 3. The experimental results of several object detection models.

Model	Precision	Recall	F1 Score	mAP@0.5	mAP@0.5:0.95
YOLOv4	0.398	0.442	0.419	0.417	0.117
YOLOv7	0.592	0.436	0.502	0.474	0.175
Ours (YOLOv4/PRB + CA + T)	0.647	0.616	0.631	0.645	0.272
Ours (YOLOv4/PRB + CA)	0.632	0.627	0.629	0.644	0.275
Ours (YOLOv4/PRB + T)	0.646	0.593	0.618	0.626	0.262
Ours (YOLOv7/PRB + CA + T)	0.632	0.578	0.604	0.594	0.256
Ours (YOLOv7/PRB + CA)	**0.880**	0.573	**0.694**	0.631	0.252
Ours (YOLOv7/with PRB + T)	0.659	**0.636**	0.647	**0.665**	**0.310**

The proposed model architecture in this study shows improvement in all evaluation metrics compared to their original models. The larger the value, the better the performance in the above table. The boldface numbers indicate the best results of each column. It can be seen that incorporating PRB, CA, or T will significantly make the corresponding original models better in various metrics. Particularly, when considering mAP@0.5 as the main evaluation metric, our model with PRB + T outperforms the original model by more than 40% in this aspect.

Next, we conducted three control experiments to evaluate the effect of using attention mechanisms, as shown in Table 4. The evaluation metrics used for comparison were precision, recall, F1 score, and mAP for the three types of traffic signals: red, green, and yellow. The first experiment used PRB as the control without any attention mechanism. The second experiment replaced our model's transformer block with coordinate attention. The third experiment utilized the proposed transformer block architecture from this study. The results of these three experiments were compared to understand the impact of different attention mechanisms on the performance of the models.

Table 4. The experimental results of different traffic signals and attention mechanisms.

	Red	Green	Yellow
PRB			
Precision	0.776	0.838	0.576
Recall	0.561	0.809	0.786
F1 score	0.651	0.823	0.665
mAP@0.5	0.617	0.796	0.752
mAP@0.5:0.95	0.217	0.297	0.375
Ours (PRB + CA)			
Precision	0.891	**0.920**	0.710
Recall	0.549	0.809	0.935
F1 score	0.679	0.861	0.807
mAP@0.5	0.740	0.868	**0.918**
mAP@0.5:0.95:	0.269	0.351	0.387
Ours (PRB + T)			
Precision	**0.936**	0.914	**0.785**
Recall	**0.754**	**0.830**	**0.961**
F1 score	**0.835**	**0.870**	**0.864**
mAP@0.5	**0.866**	**0.881**	0.913
mAP@0.5:0.95	**0.404**	**0.430**	**0.404**

According to the experimental results, the proposed architectures PRB + T and PRB + CA both show improvements over the PRB architecture in all overall evaluation metrics. The boldface numbers indicate the best results for each metric in all of the models: PRB, PRB + CA, and PRB + T.

To demonstrate that the model with the self-attention mechanism is indeed better suited for training with the Adam optimizer, we conducted experiments to confirm whether the model would achieve better results after using the Adam optimizer. We compared two models that include self-attention mechanisms and used two optimizers for comparison. The experimental results are shown in Table 5.

Table 5. The experimental results of self-attention and optimizers.

Model	Classes	Precision	Recall	F1 Score	mAP@0.5	mAP@0.5:0.95
E-ELAN + PAN + T with SGD	All	0.529	0.558	0.543	0.540	0.202
	Red	0.794	**0.788**	0.791	0.800	0.284
	Green	0.750	0.816	0.782	0.803	0.305
	Yellow	0.574	0.630	0.601	0.559	0.220
E-ELAN + PAN + T with Adam	All	0.604	0.578	0.591	0.609	0.286
	Red	0.912	0.637	0.750	0.809	0.306
	Green	**0.934**	0.810	0.868	**0.881**	0.420
	Yellow	0.568	0.864	0.685	0.742	**0.419**
E-ELAN + PRB + T with SGD	All	0.576	0.541	0.558	0.557	0.203
	Red	0.739	0.750	0.744	0.741	0.246
	Green	0.782	**0.849**	0.814	0.829	0.278
	Yellow	0.783	0.565	0.656	0.659	0.287
E-ELAN + PRB + T with Adam	All	**0.659**	**0.636**	**0.647**	**0.665**	**0.310**
	Red	**0.936**	0.754	**0.835**	**0.866**	**0.404**
	Green	0.914	0.830	**0.870**	**0.881**	**0.430**
	Yellow	**0.785**	**0.961**	**0.864**	**0.913**	0.404

The above experimental results confirm that the models containing the self-attention mechanism are better suited for using the Adam optimizer, as they show further improvements in various evaluation metrics. The boldface numbers indicate the best results for each metric and each traffic light color under the four combinations of model and optimizer.

4.4. Ablation Studies

The ablation experiments in this research aim to validate the impact of different modules on the model's capabilities and performance. We conducted three types of table experiments, including ablation experiments, parameter quantity experiments, and module replacement experiments.

In these experiments, we introduced another type of Backbone and tried adding only one method to the model, as well as a combination of methods like coordinate attention, transformer block, etc., to compare whether the model's capabilities could be further improved. The results are shown in Table 6.

From the above experimental results, we observed that within the E-ELAN framework, replacing PRB + T with PRB + CA resulted in slightly higher precision and F1 score, but when both PRB + CA and transformer block were used together, PRB + CA + T, the overall performance decreased. The boldface numbers indicate the best results for each metric under the two basic models with different components. Among the models without using PRB, PAN + CA achieved slightly higher mAP@0.5:0.95compared to the PRB + T model. Although PRB + T did not achieve the highest scores in two of the metrics, it outperformed other models in recall and mAP@0.5. The differences in the remaining evaluation metrics were also minor, indicating that the use of PRB in combination with transformer block in this research is highly capable of object detection and recognition tasks.

Table 6. The experimental results of ablation study.

Basic Model	Component	Precision	Recall	F1 Score	mAP@0.5	mAP@
CSPDarknet53	PAN + CA	0.614	0.603	0.608	0.629	0.265
	PAN + T	0.606	**0.635**	0.620	0.643	**0.296**
	PRB	0.583	0.486	0.530	0.508	0.178
	PRB + CA + T	**0.647**	0.616	**0.631**	**0.645**	0.272
	PRB + CA	0.632	0.627	0.629	0.644	0.275
	PRB + T	0.646	0.593	0.618	0.626	0.262
E-ELAN	PAN + CA	0.657	0.630	0.643	0.653	**0.315**
	PAN + T	0.604	0.578	0.591	0.609	0.286
	PRB	0.548	0.539	0.543	0.541	0.222
	PRB + CA + T	0.632	0.578	0.604	0.594	0.256
	PRB + CA	**0.880**	0.573	**0.694**	0.631	0.252
	PRB + T	0.659	**0.636**	0.647	**0.665**	0.310

Moving on to the experiments based on the CSPDarknet53 architecture, we found that the model with the best performance was different from that in E-ELAN. Although it did not surpass the results obtained with E-ELAN, in CSPDarknet53, PAN + T, and PRB + CA + T showed the best results. These findings demonstrate that we can modify the architecture according to the characteristics of the different base frameworks. Depending on the specific base framework and its properties, we can adjust to achieve the most optimal results.

To better understand the differences in the detection results of each model, we present images from the test dataset for comparison. The images are shown in Figure 6. As the traffic lights in this dataset are quite small, we first display the original images and then overlay the results of each model after cropping them for better visualization.

Original image (**a**) PRB + CA

Figure 6. *Cont.*

(**b**) PRB+ (**c**) PAN + CA

Figure 6. The sample results of different models. (**a**) PRB + CA; (**b**) PRB + T; (**c**) PAN + CA.

Based on the results shown in the images above, we observed the following:

In Figure 6a, PRB + CA correctly detected all the traffic lights, but there was a tendency to neglect the traffic lights on the left side of the image; In Figure 6b, PRB + T the model ultimately chosen in this study, accurately detected all the traffic lights without any attention errors. It demonstrated superior stability and correctness compared to other models; In Figure 6c, PAN + CA successfully recognized the traffic lights at the front intersection, but it tended to overlook the smaller traffic lights approaching the next intersection.

Incorporating the PRB component or attention block into the basic model, we observe that the number of parameters does not increase much. For example, the model size of the model CSPDarknet53 is 52 M; when replacing PAN with PRB, the model size increases to 61 M, and when replacing PAN with PRB + CA + T, the model size increases to 66 M. Note that, applying the same replacements to the model E-ELAN, the model size is increased on a slightly larger scale.

5. Conclusions

We proposed a model architecture that incorporates the methods and concepts of PRB, ViT, and CA, and combine their respective advantages and characteristics, maximizing their effectiveness in suitable positions. Through the experimental results of detecting traffic lights, we observed that our model architecture improved the evaluation metrics, including precision, recall, F1 score, mAP@0.5, and mAP@0.5:0.95. This indicates that our model achieved significant progress in both detection and classification tasks, which is crucial for fulfilling the requirements of scene detection and scene recognition in autonomous driving technology. The results of the ablation experiments further confirmed that all the proposed modules and methods contributed to the improvement of the model's capabilities. Moreover, we validated the effectiveness of PRB's methods, demonstrated the new application scenarios proposed by ViT, and showed how the CA resulted in an enhanced architecture that improved the model's capabilities, especially in handling fine-grained

objects. However, separate methods are used to extract effective features for object detection in this study, and developing a unified method may be a future direction. Furthermore, applying lightweight architectures for edge computing deserves further study.

Author Contributions: Conceptualization, C.-H.C., C.-C.L., J.-H.L. and K.-C.F.; methodology, C.-H.C., C.-C.L., J.-H.L. and K.-C.F.; software, C.-H.C. and C.-C.L.; validation, C.-H.C. and C.-C.L.; investigation, C.-H.C.; data curation, C.-H.C.; writing—original draft preparation, C.-H.C.; writing—review and editing, C.-H.C. and C.-C.L.; visualization, C.-H.C.; supervision, J.-H.L. and K.-C.F.; project administration, J.-H.L. and K.-C.F.; funding acquisition, J.-H.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Science and Technology Council of Taiwan, under grant no. [MOST 111-2221-E-431-001].

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

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

Abbreviations

CNN	Convolutional neural networks
FPN	Feature pyramid network
PRB-FPN	Parallel residual bi-fusion feature pyramid network
PRB	PRB-FPN
CA	Coordinate attention
T	Transformer block
ViT	Visual transformer
P	Precision
R	Recall
AP	Average precision
mAP	Mean average precision
PANet	Path aggregation network
PAN	PANet
YOLO	You only look once
SSD	Single shot detector
NMS	Non-maximum suppression

References

1. SAE Standards News: J3016 Automated-Driving Graphic Update. Available online: https://www.sae.org/news/2019/01/sae-updates-j3016-automated-driving-graphic (accessed on 9 July 2023).
2. Masaki, S.; Hirakawa, T.; Yamashita, T.; Fujiyoshi, H. Distant Traffic Light Recognition Using Semantic Segmentation. *Transp. Res. Rec.* **2021**, *2675*, 97–103. [CrossRef]
3. Niu, C.; Li, K. Traffic Light Detection and Recognition Method Based on YOLOv5s and AlexNet. *Appl. Sci.* **2022**, *12*, 10808. [CrossRef]
4. Greer, R.; Gopalkrishnan, A.; Landgren, J.; Rakla, L.; Gopalan, A.; Trivedi, M. Robust Traffic Light Detection Using Salience-Sensitive Loss: Computational Framework and Evaluations. *arXiv* **2023**, arXiv:2305.04516.
5. Bochkovskiy, A.; Wang, C.-Y.; Liao, H.Y.-M. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv* **2020**. [CrossRef]
6. Wang, C.-Y.; Bochkovskiy, A.; Liao, H.-Y.M. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *arXiv* **2022**. [CrossRef]
7. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. Ssd: Single shot multibox detector. In Proceedings of the European Conference on Computer Vision (ECCV), Amsterdam, The Netherlands, 8–16 October 2016; pp. 21–37.
8. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *arXiv* **2016**. [CrossRef]
9. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. *arXiv* **2018**. [CrossRef]
10. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. *arXiv* **2016**. [CrossRef]
11. Redmon, J.; Farhadi, A. YOLO9000: Better, Faster, Stronger. *arXiv* **2016**. [CrossRef]
12. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. *arXiv* **2018**. [CrossRef]
13. Yolov5. Available online: https://github.com/ultralytics/yolov5 (accessed on 9 July 2023).

14. Li, C.; Li, L.; Jiang, H.; Weng, K.; Geng, Y.; Li, L.; Ke, Z.; Li, Q.; Cheng, M.; Nie, W.; et al. YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications. *arXiv* **2022**. [CrossRef]
15. Ding, X.; Zhang, X.; Ma, N.; Han, J.; Ding, G.; Sun, J. RepVGG: Making VGG-style ConvNets Great Again. *arXiv* **2021**. [CrossRef]
16. Weng, K.; Chu, X.; Xu, X.; Huang, J.; Wei, X. EfficientRep: An Efficient Repvgg-style ConvNets with Hardware-aware Neural Network Design. *arXiv* **2023**. [CrossRef]
17. Ultralytics. Available online: https://github.com/ultralytics/ultralytics (accessed on 9 July 2023).
18. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature Pyramid Networks for Object Detection. *arXiv* **2017**. [CrossRef]
19. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path Aggregation Network for Instance Segmentation. *arXiv* **2018**. [CrossRef]
20. Chen, P.-Y.; Chang, M.-C.; Hsieh, J.-W.; Chen, Y.-S. Parallel Residual Bi-Fusion Feature Pyramid Network for Accurate Single-Shot Object Detection. *IEEE Trans. Image Process.* **2021**, *30*, 9099–9111. [CrossRef] [PubMed]
21. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *arXiv* **2017**, arXiv:1706.03762.
22. Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; Zagoruyko, S. End-to-End Object Detection with Transformers. *arXiv* **2020**. [CrossRef]
23. Lee, Y.; Hwang, J.-W.; Lee, S.; Bae, Y.; Park, J. An Energy and GPU-Computation Efficient Backbone Network for Real-Time Object Detection. *arXiv* **2019**. [CrossRef]
24. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *arXiv* **2020**. [CrossRef]
25. Aksoy, T.; Halici, U. Analysis of visual reasoning on one-stage object detection. *arXiv* **2022**. [CrossRef]
26. Hu, J. Squeeze-and-Excitation Networks. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018.
27. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. CBAM: Convolutional Block Attention Module. *arXiv* **2018**. [CrossRef]
28. Hou, Q.; Zhou, D.; Feng, J. Coordinate Attention for Efficient Mobile Network Design. *arXiv* **2021**. [CrossRef]
29. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2017**. [CrossRef]
30. Ruder, S. An overview of gradient descent optimization algorithms. *arXiv* **2017**. [CrossRef]
31. Bosch-Ros-Pkg/Bstld. Available online: https://github.com/bosch-ros-pkg/bstld (accessed on 9 July 2023).