SelectBlur: A Pre-Processing Algorithm for High-Performance Vehicle Detection at Night in Fisheye Surveillance Videos

Shafin Bin Hamid, Himaddri Roy, Munshi Sanowar Raihan, Prasun Datta, Ashiqur Rasul, Mohammad Ariful Haque

***Abstract*—Traffic surveillance and monitoring using fisheye cameras are gaining popularity because of the 360-degree wide-angle view. The accuracy of vehicle detection from fisheye images has been significantly improved by using high-performance deep learning-based object detectors. However, one key area of improvement remains in detecting vehicles from night-time images due to the low brightness and contrast against the background. We have found that it is possible to make night images suitable for the model to learn from by intentionally blurring out selective portions of the images before training. In our proposed technique termed as SelectBlur, we first divide a night image into square grids and depending on whether a grid meets certain conditions, we decide on blurring it. It is shown that blurring out parts of the image that are known to not contain any vehicles leads to significantly improved performance. The SelectBlur, in conjunction with state-of-the-art object detectors, such as Yolov5x and Yolov5s, beats the baseline model without pre-processing by 5.3% and 3.0% improvement in mean average precision, respectively. An ablation study of our proposed algorithm is also performed by considering different conditions for blurring, and also varying the type and the size of the kernel used to perform the blurring operation.**

***Index Terms*—Vehicle Detection, Night Surveillance, Fisheye, Pre-processing, SelectBlur, Yolov5**

I. Introduction

VEHIcle detection is an important task in the context of both research and industrial applications. In the age of intelligent transportation systems, vehicle detection plays a crucial part in acquiring information about several significant details such as the count of vehicles passing through a junction, their speed, the distance between vehicles, forecasting traffic at a certain time based on previous events, etc [1], [2], [3], [4]. Recently, fisheye cameras have gained a lot of popularity in applications such as vehicle detection at traffic junctions. This can be largely attributed to the fact that a fisheye lens provides a wide-angle view of the junction and therefore covers a larger footprint compared to a conventional pinhole camera. As a result, the number of cameras needed for surveillance at a single junction is reduced and fisheye cameras serve as a low-cost alternative [5], [6].

Due to the recent advances in deep learning, object detection has reached newer heights. Recent surveys conducted by [7], [8] summarize the evolution of research in the field of object detection over the years, starting from the application of hand-crafted feature extractors to deep neural networks(DNN). At present, state-of-the-art object detectors based on DNNs can be broadly classified into two categories, multi-stage detectors and single-stage detectors. Multi-stage detectors [9], [10], [11] use a separate region proposal network (RPN) that hypothesizes regions of interest in the whole image, and those regions are then sent to the detector head responsible for classification and localization. On the other hand, single-stage detectors [12], [13], [14] use an end-to-end training scheme where predetermined anchors placed on an image are directly regressed to object boundaries. Initially, the trade-off between the two kinds of detectors was between high accuracy and high speed. While the two-stage networks were more accurate in detection performance, their single-stage counterparts compensated that with a comparatively higher inference speed. But as the research in this field matured, the single-stage detectors overcame the initial low performance in accuracy. State-of-the-art object detectors such as Yolov5 currently provide detection accuracy on par with two-stage detectors while maintaining the real-time inference speed [15].

A lot of research has already gone into using fisheye images for different purposes, e.g object detection, semantic segmentation. Although there are certain advantages of using fisheye cameras in such applications, there are some question marks due to the high distortion effects introduced by the fisheye lens[16], [17]. Notable progress made in solving this problem ranges from the invention of new augmentation techniques designed to handle the distortion effect [18], [19] to using deformable convolution layers for extracting features in fisheye images [20], [21].

In the context of vehicle detection from fisheye images, the most significant works done so far use surround view cameras placed on self-driving cars [22], [23]. And there is a lack of availability of large-scale annotated dataset for vehicle detection in surveillance applications. The dataset that we have presented our work on made its debut in the IEEE Video and Image Processing Cup 2020, and it consists of a number of surveillance videos collected by fisheye cameras mounted on street lamps. There are a larger number of vehicles of varying sizes in the same frame as compared to what there would be in pictures taken from a surround view camera, thereby making it a more challenging problem.

After training Yolov5 directly on this dataset, we have observed that the performance is sub-optimal when it comes to

The authors are with the Department of Electrical and Electronic Engineering, Bangladesh University of Engineering and Technology, Dhaka 1205, Bangladesh (e-mail: arifulhoque@eee.buet.ac.bd).

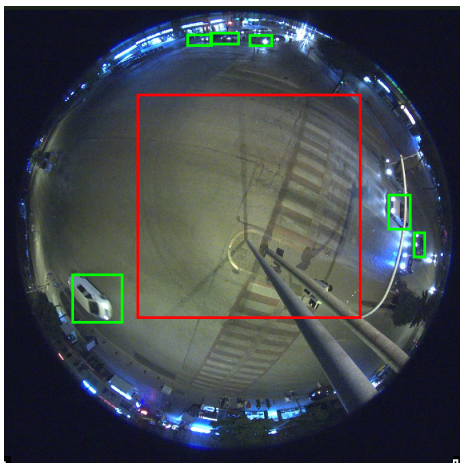


Fig. 1: Example of a frame in a night scene. The Green rectangles show the bounding boxes for vehicles. The red rectangle in the middle represents a large region with no useful information.

detecting vehicles in night images. A closer look at a typical night image in Fig 1 tells us that there are certain areas in it that do not contain any vehicles, as illustrated by the red rectangle. If we can reduce the amount of information content in the area inside the red rectangle, then our model will de-emphasize that region and focus more on the areas that contain vehicles. In that regard, we have proposed an algorithm termed as SelectBlur to select the regions in a night image that do not contain vehicles. Then we have made use of low-pass filters, such as box filters and Gaussian filters to blur those irrelevant portions in the image. Experimental results show that after using our selective blur method to pre-process images before training, there is a performance improvement of as high as 5.3% mAP for the large model, Yolov5x and 3.0% mAP for the small model, Yolov5s.

The rest of the paper is organized as follows: In the next section, we describe the prior work done with regard to vehicle detection in night scenes and the usage of conventional signal processing algorithms to pre-process images before training a deep neural network. In Section III, we thoroughly discuss our algorithm to selectively blur parts of an image and the method of dataset preparation for training. In Section IV, a more detailed description of the dataset is provided along with the training configurations for our experiments. Finally, in Section V, we present the experimental results generated by testing our model on the night scenes.

II. Related Works

In the recent few years, there have been significant amounts of work on solving the detection problem in night vision. Difficulty in detecting objects at night increases because firstly, night-time training data is scarce, and secondly, night-time images have low illumination and texture properties. To overcome these challenges, some researchers have built a contrast model where the object boundaries are detected by extracting local contrast and entropy values from each image and also correlating the detection from two consecutive frames [24], [25]. This method based on handcrafted features cannot perform as accurately as current deep CNNs used in object detection tasks. Some deep-learning based models have been developed especially to handle night-time images. For instance, a segmentation model accompanied with Generative adversarial networks (GAN) has been built up to perform inference directly from night images in [26]. The authors have applied GAN to generate night images from the available day-image dataset and vice-versa. In case of the night-trained model, the day-to-night converted images have been fed as input while taking advantage of the labels and for the day-trained model, night-to-day conversion has been conducted in the inference stage to obtain better performance from the model. A similar kind of semantic gap reduction process has been found in the work of Dengxin *et al.* [27], where the model's knowledge trained on daytime images has been transferred to night-time images by subsequent distillation through civil, nautical, and astronomical twilight time images. Also, in [28], the authors generated virtual target scenes similar to the daytime environment through game training of generators and discriminators using Deep Convolutional Generative Adversarial Network (DCGAN) and later combined with the Faster R-CNN target detection system through deep convolutional feature fusion and multi-scale Region Of Interest(ROI) pooling. Although making use of deep generative networks has provided better results than previously used handcrafted features, training a GAN still remains a cumbersome task due to the unsupervised learning method.

The focal point of our work is to pre-process the night images by filtering the unnecessary regions before sending them to the network during training. We have found numerous studies where images are first pre-processed using conventional signal processing algorithms to make them more suitable for training deep neural networks. Kvyetnyy *et al.* [29] proposes image preprocessing and denoising through bilateral filtering and wavelet thresholding and makes use of modified Haar-like features and symmetrical local binary patterns. Yuen *et al.* [30] also recommends image enhancing with localized enhancement functions for regions, utilizing Gaussian processes to govern them into a distribution of functions and feeding these enhanced images into training a neural network.

III. Proposed Methodology

Our method for blurring night images before using them for training can be predominantly split up into two segments. The first one is the *estimation* or figuring out which portions of the image we can and cannot blur. And the second segment is the *blur mechanism*, or the technique followed to blur the portions. In the following two subsections, we sequentially demonstrate these two segments and their effects on the training images.

A. Estimation

1) Motivation: One of the characteristics we have noticed in almost every moving vehicle in night-time images is that they have their front headlights and rear headlights on. This feature can facilitate us in determining whether we have a car

in a certain segment of an image. We will now elucidate this by the set of images shown in Fig 2.

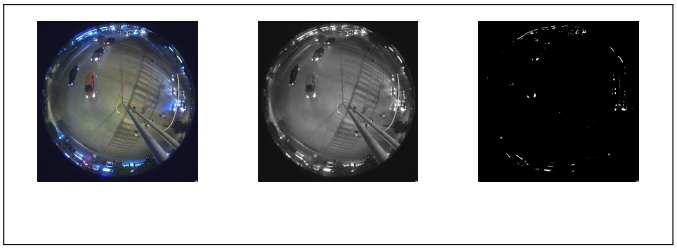


Fig. 2: The original image (leftmost) is first grayscaled (middle) and then binarized (rightmost) using a threshold. Significant parts of the binary image are black and no vehicle is present there effectively making the regions irrelevant for our detector.

Here, the leftmost image is a typical example of a frame at night with a number of vehicles, all of which have their headlights on. The image in the middle is the grayscale version of the same image and the one on the right is the result of binarizing the grayscale image. A threshold τ was used to carry out the binarizing operation on the grayscale image, denoted by *GrayImage* and the result was the binary image, denoted by *BinImage*. The pixel intensities in the grayscale image were normalized and τ was set to 0.95 for our application. The method is summarized in (1).

$$BinImage(i,j) = \begin{cases} 1, & \text{if } GrayImage(i,j) < \tau. \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

The binarized image reveals a few interesting details while also throwing a few challenges. Firstly, we can clearly see the headlights of the cars as white regions against the background. But the difficulty arises when we try to distinguish these headlights from lights arriving from elsewhere in the image e.g. the shops on the roadside. It is, therefore, safer to leave those portions as they are, because any attempt to blur the roadside lights can result in neglecting vehicles that are close to the shops and end up hurting our detector's performance. But the main takeaway from the binarized image in Fig 2 remains that we can, in fact, recognize large portions of the image which are entirely black and do not contain any useful information.

2) Algorithm: The white regions that indicate headlights in the binarized image appear randomly. There is no way to accurately guess how the car is positioned from those regions. So, we decided to divide the image into 16 (4 rows, 4 columns) equally sized grids, as shown in Fig 3 to keep the uniformity. The grids were kept large enough so they can encompass a whole vehicle.

Now, for every grid, we need to devise a uniform constraint by which we can decide on whether that grid contains useful information. To do that, we first find out the regions of interest (ROI) in the binary image which in our case are the white blobs on the black background. Based on the number of ROIs detected and the area of each of them, we devise two different sets of constraints on the decision to blur that grid. The reasoning behind each of the constraints is discussed below:

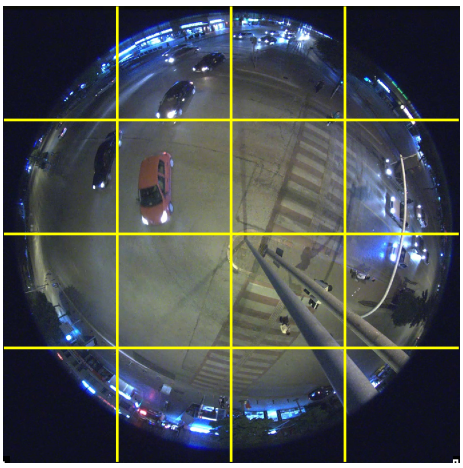


Fig. 3: Image divided into 4 $\times$ 4 uniform grids. The target is to selectively blur some of these grids by imposing some constraints.

Set of Constraints, C1: If any grid in the image is entirely black, then we blur that grid. Although the chances of blurring a grid with a vehicle in it reduce to almost zero after imposing this constraint, there are still large portions of the image containing no vehicles but they remain unchanged.

Set of Constraints, C2: We will first disregard the ROIs having an area less than 4 pixels since they represent noisy white blobs, shown in Fig 4. After that, if a grid contains less than 4 ROIs and all of them have an area less than 100 pixels, then we blur that grid. The reason for having an additional constraint on the area of ROIs is to identify headlights in a single grid, as illustrated by Fig 5.

Fig. 4: The grid on the left does not contain any light of significance, yet the binarized image reveals a number of noisy white blobs that should not be considered as separate ROI's.

Our method for finding out which portions to blur in an image is recapped in **Algorithm 1**. In essence, we are relaxing the constraints progressively from Sets 1 to 2, due to which more square grids are getting blurred, as illustrated in Fig 6, We have summarized our results due to blurring using different constraints in the following sections.

B. Blur Mechanism

1) Normalized Box Blur: The box blur method uses a square kernel that averages all the pixels in an N $\times$ N

Algorithm 1: SelectBlur Pre-Processing

Data: TrueImage, TI
Result: BlurredImage, BI
/*Read an image, calculate size of square grid based on width of image*/

[Width, Height] = $size$(TI)
GridSize = Width/4
/*Grayscale and binarize*/
GrayImage = $Convert2Gray$(InputImage)
BinImage = $Binarize$(GrayImage, 0.95)
/*Iterating over all grids*/
for $i \leftarrow 1$ *to* 4 **do**
  for $j \leftarrow 1$ *to* 4 **do**
    /*Getting Current Grid*/
    CurrentGrid = $SelectGrid$(BinImage(i, j))
    /*Generating the ROI's*/
    ROI = $DetectROI$(CurrentGrid)
    AreaROI = $CalculateArea$(ROI) /*area of ROI's*/
    if *isempty(AreaROI)* **then**
      ToBlur = 1 //constraint 1
    end
    else
      index = $find$(AreaROI>=4)
      AreaROI = AreaROI(index)
      if *length(AreaROI)<4 & max(AreaROI)<100* **then**
        ToBlur = 1 //constraint 2
      end
    end
    else
      ToBlur = 0
    end
    if *ToBlur* **then**
      /*Blur operation*/
      CurrentGrid = $Blur$(CurrentGrid)
    end
  end
  $SelectGrid$(BI(i,j)) = CurrentGrid
end

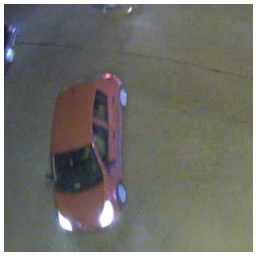
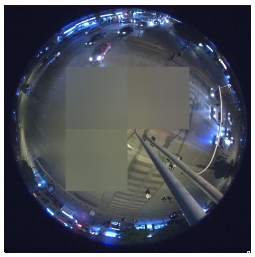
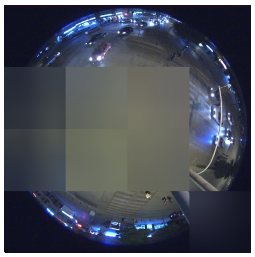


Fig. 5: The original grid (left) contains one vehicle and the binarized grid (right) shows the two headlights only as the two ROI's. To make sure this grid does not get blurred, an additional constraint is imposed such that any ROI having an area greater than 100 pixels is considered as a headlight and the grid containing that ROI is left unchanged.



(a) Blurred image under C1

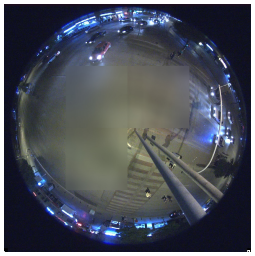


(b) Blurred image under C2

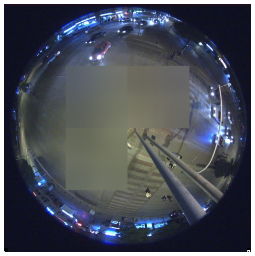
Fig. 6: Difference between blurred image under set of constraints, C1 and C2. The grids that are blurred under C1 are also blurred under C2, but C2 contains more blurred grids.

neighborhood (N is the width of the kernel) and replaces the center pixel with that value, as represented by (2):

$$F(u,v) = \frac{1}{N^2}[\sum_{i=-(N-1)/2}^{(N-1)/2} \sum_{j=-(N-1)/2}^{(N-1)/2} I(u+i, v+j)] \quad (2)$$



(a) Kernel Size = 51×51



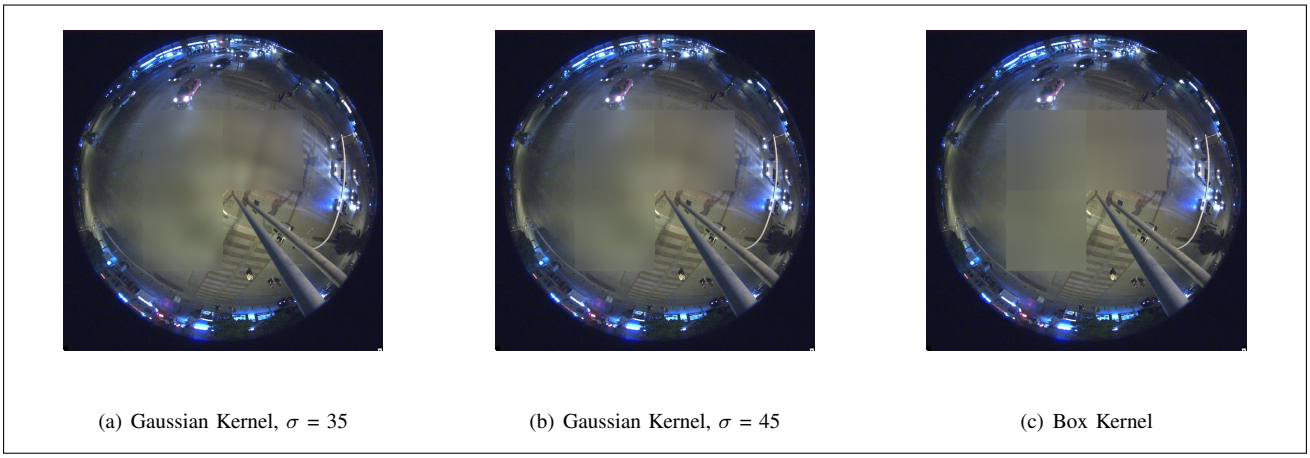
(b) Kernel Size = 201×201

Fig. 7: Effect of increasing kernel size from 51×51 to 201×201 for normalized box blur method. As kernel size is increased, the level of blur in a grid becomes more observable indicating further deviation from the original grid.

The target is to reduce the amount of information in a particular grid by averaging neighborhood pixels using the box filter. The by-product of increasing the kernel width is reflected in Fig. 7. It is evident that increasing the kernel width will only make the blurred image deviate more from the original image due to the contribution from more neighborhood pixels. The kernel width is intentionally kept odd to better specify the position of the center pixel.

2) Gaussian Blur: A normalized box filter or the mean blur is the most aggressive procedure to blur an image since it assigns the same weight to the corner pixels as it does to the center pixel. An example of a more controlled and edge-preserving blur mechanism is the Gaussian blur.

The principle of blurring is similar to the mean blur, as represented by (3):



(a) Gaussian Kernel, σ = 35 (b) Gaussian Kernel, σ = 45 (c) Box Kernel

Fig. 8: Effect of increasing the standard deviation, σ of the Gaussian kernel on the blurred image keeping kernel size fixed at 201×201. As σ increases, the effect of blurring becomes more evident and the resulting image approximates the blurred image using a box kernel.

$$F(u,v) = \frac{1}{N^2}[\sum_{i=-(N-1)/2}^{(N-1)/2}\sum_{j=-(N-1)/2}^{(N-1)/2} I(u+i,v+j)*H(i,j)] \tag{3}$$

where the weights of the Gaussian filter are governed by (4):

$$H(i,j) = \frac{1}{2\pi\sigma^2} \times exp[-\frac{i^2+j^2}{2\sigma^2}] \tag{4}$$

Here, σ represents the standard deviation of the Gaussian kernel. As the standard deviation of the kernel is increased, the radius of the Gaussian bell curve becomes wider. So, the weights at the same distance from the mean will be greater than what it previously was for a smaller σ. The effect of changing the standard deviation is further explained by Fig. 8. It is clear that the resulting image will approximate the original image if σ is decreased. And, it will approximate the blurred image after using the box kernel if σ is increased.

IV. Experimental Setup

A. Dataset Description

In table I, we have summarized the basic information of all the videos. We use the dataset in three different modes. First, we use the original frames for both training and testing. Then, we only train our network with blurred images and test with original images. And, finally we blur the frames for both training and testing.

TABLE I: Dataset Summary

Data Classification	Video Sequence Names (Frame size)	No. of Frames
Train	CLIP_20200610-213240 (1024*1024)	2350
	CLIP_20200610-213821 (1280*1280)	2275
	CLIP_20200610-214657 (1280*1280)	4811
Test	CLIP_20200628-210253 (1056*1056)	554
	CLIP_20200610-210808 (1056*1056)	612

B. Training Configurations

We decided to train our networks using different architectures, combinations of constraints applied on the image grids, kernel width, and blurring mechanism. A summary of the training scheme is presented below:

- Model architectures: We have used Yolov5 as our baseline object detector. The model architecture is shown in Fig 9. We have used two variants of this model, namely Yolov5x and Yolov5s. The former is the larger of the two models having over 10 times as many parameters. This yields better detection performance for the bigger model although at the expense of a significantly larger inference time.
- Constraints on selective blur: We have applied two different sets of constraints, C1 and C2, to selectively blur portions in a night image as discussed in the previous section.
- Kernel width: We have experimented with two different kernel sizes, 51×51 and 201×201 for blur operation.
- Blur mechanism: To vary the algorithm used to blur the image, we have used box blur and Gaussian blur with two different standard deviations for each kernel size. We chose σ = 5, 15 for 51 × 51 sized kernel, and σ = 35, 45 for 201 × 201 sized kernel. The standard deviation is increased with kernel size. Otherwise, the corner pixels will be weighted so low that they will make virtually no difference as compared to blurring with a smaller-sized kernel.

The reason behind training using different combinations is to achieve optimal performance as well as understand if there are any clear and obvious trends in the performance with respect to varying any of the three parameters. The different training configurations along with their results are presented in the next section.

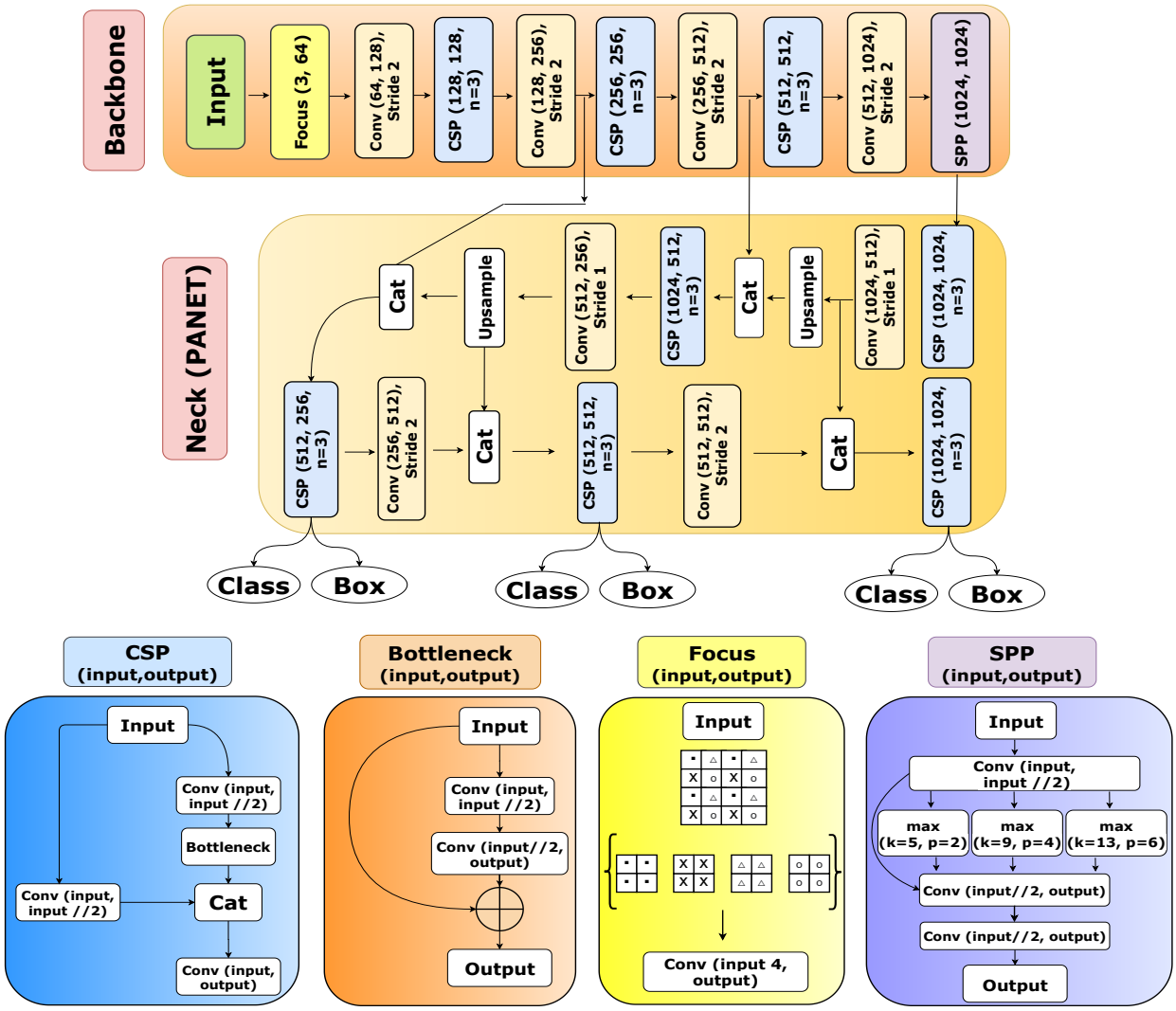


Fig. 9: Architecture of Yolov5 model. To get the small model (Yolov5s), the shown depth and width are multiplied by 1/3 and 1/2 respectively. In the case of the extra-large model (Yolov5x), the depth and the width multiples are 4/3 and 5/4 respectively. The design of Focus, CSP, and SPP blocks are shown at the bottom.

V. Results and Analysis

To evaluate the detection performance, we have used mean average precision (mAP) as our performance metric. Our target is to improve detection performance on night scenes in particular. As mentioned in the previous section, we have used the whole dataset in three different modes and the best results in those three modes are summarized in table II.

TABLE II: Best Performance for Different Modes of Data.

Training	Testing	Yolov5x (%mAP)	Yolov5s (%mAP)
W/O Preprocessing	W/O Preprocessing	45.9	41.8
SelectBlur	W/O Preprocessing	50.2	44.3
SelectBlur	SelectBlur	**51.2**	**44.8**

When we train the network with raw night images, the mAP we get is 45.9% for the large Yolov5x model and 41.8% for the small Yolov5s model. In comparison, when we use SelectBlur to pre-process both the training and test images before feeding them to the network, the mAP is 51.2% for YoloV5x and 44.8% for YoloV5s. This shows that our proposed method of selectively blurring parts of the images before feeding them to our model outperforms the baseline method of training with raw images by 5.3% mAP for Yolov5x and 3.0% mAP Yolov5s. The results show that even if we use SelectBlur to pre-process the training images only, the mAP is 50.2% for Yolov5x and 44.3% mAP for Yolov5s, that are significantly better than the baseline results.

We will now present a detailed ablation study of our training configuration by taking various combinations of constraints, blur mechanisms, and kernel sizes for both our model architectures. When we change one, we will keep the other two fixed so as to better understand the effect of tuning that parameter alone. All the results hereafter are based on blurring the test images at inference since we have already seen that blurring test images leads to better performance. It should be mentioned that while blurring at inference, we are using the

same combination of constraint, mechanism, and kernel size as we did in training making our network completely end-to-end.

A. Changing constraints keeping a fixed kernel size and blur mechanism

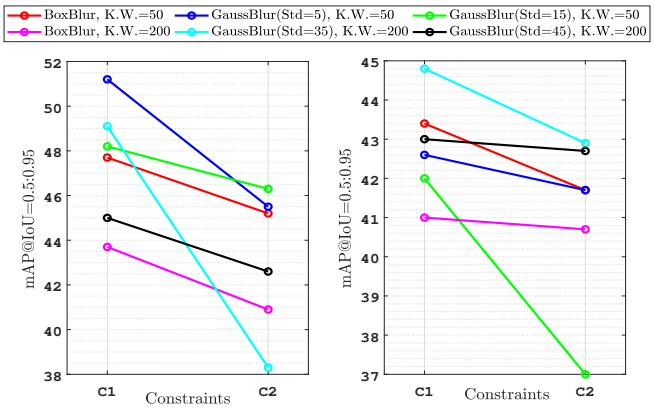


Fig. 10: Effect of changing constraint applied

The results shown in Fig.10 display the changes of mAP values from constraint C1 to constraint C2 for all the combinations of blurring mechanisms and kernel sizes. From these figures, it is evident that constraint, C1 is performing better than constraint, C2 for most of the combinations. For Yolov5s, the trend is not as obvious whereas, for Yolov5x, C1 outperforms C2 in each case. This basically means that blurring only those grids that are entirely black leads to better performance than relaxing the constraints further and blurring more grids.

B. Changing blur mechanism keeping a fixed kernel size and constraint

Fig.11 shows the effects of changing the blurring mechanism from the Gaussian blur method to the normalized box method. There are no clear and obvious trends this time as there were in the case of changing constraints. However, we can infer from the plots that Gaussian blurring with low standard deviation performs better in the majority of cases.

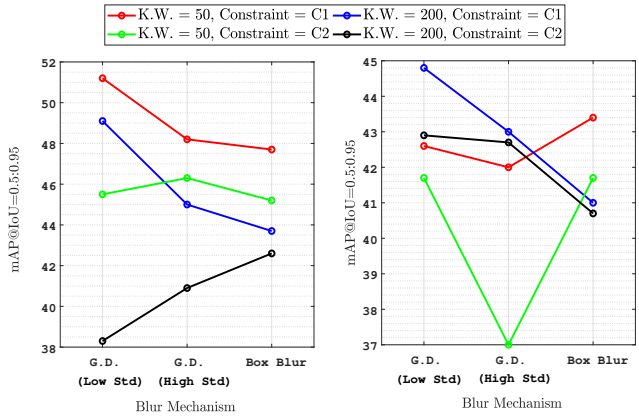


Fig. 11: Effect of changing blur mechanism

C. Changing kernel size keeping a fixed blur mechanism and constraint

The effects of changing the kernel size on the resulting mAPs are shown in Fig.12. For Yolov5x, increasing the width of the kernel from 51 to 201 leads to a drop in performance in all cases. However, for Yolov5s, the effect is opposite and mAP increases when we increase the kernel size for low and high standard deviations and mAP decreases with the increase in kernel size for box blur. This leads us to believe there is a standard deviation over our chosen high σ after which the mAP starts to drop with the increase in kernel size.

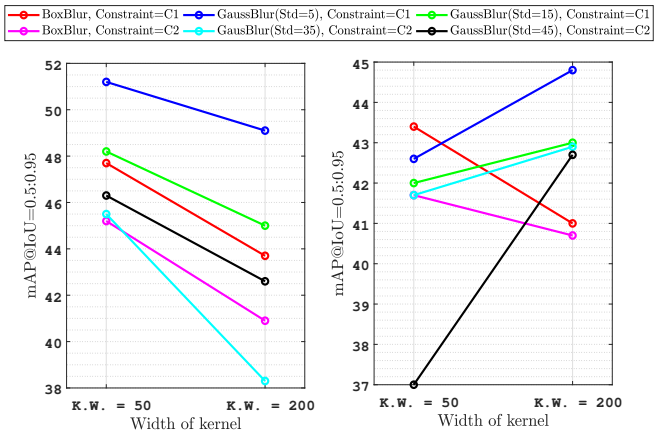


Fig. 12: Effect of changing kernel size

VI. Conclusion

In this paper, we have focused on improving the quality of vehicle detection in fisheye surveillance videos at night. We have seen that the performance of state-of-the-art object detector Yolov5 is suboptimal in night scenes. So, we have proposed a pre-processing algorithm, SelectBlur, that works in conjunction with the Yolov5 object detector. The key idea is to reduce the amount of irrelevant information in the frames before sending them to the network for training. To do that, we use SelectBlur to find regions in frames that are potentially empty and then we blur them using conventional signal processing methods. Our experimental results show that using SelectBlur at both training and inference produces the best performance on night images. Using SelectBlur together with the conventional Yolov5 object detector beats the standalone network by 5.3% mAP for the large Yolov5x model and 3.0% mAP for the small Yolov5s model. This shows that we have achieved a significant improvement in detection performance on night images while adding only minimal computational overhead.

References

[1] J. E. Espinosa, S. A. Velastín, and J. W. Branch, "Detection of motorcycles in urban traffic using video analysis: A review," *IEEE Trans. Intell. Transp. Syst.*, pp. 1–16, 2020.

[2] H. Wu, X. Zhang, B. Story, and D. Rajan, "Accurate vehicle detection using multi-camera data fusion and machine learning," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 3767–3771.

[3] G. Wang, D. Xiao, and J. Gu, "Review on vehicle detection based on video for traffic surveillance," in *2008 IEEE International Conference on Automation and Logistics*, 2008, pp. 2961–2966.

[4] B. S. Shobha and R. Deepu, "A review on video based vehicle detection, recognition and tracking," in *2018 3rd International Conference on Computational Systems and Information Technology for Sustainable Solutions (CSITSS)*, 2018, pp. 183–186.

[5] W. Wang, T. Gee, J. Price, and H. Qi, "Real time multi-vehicle tracking and counting at intersections from a fisheye camera," in *2015 IEEE Winter Conference on Applications of Computer Vision*. IEEE, 2015, pp. 17–24.

[6] S. Li and Y. Hai, "Easy calibration of a blind-spot-free fisheye camera system using a scene of a parking space," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 1, pp. 232–242, 2010.

[7] Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 11, pp. 3212–3232, 2019.

[8] L. Jiao, F. Zhang, F. Liu, S. Yang, L. Li, Z. Feng, and R. Qu, "A survey of deep learning-based object detection," *IEEE Access*, vol. 7, pp. 128 837–128 868, 2019.

[9] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[10] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *arXiv preprint arXiv:1506.01497*, 2015.

[11] Z. Cai and N. Vasconcelos, "Cascade r-cnn: Delving into high quality object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6154–6162.

[12] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[13] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*. Springer, 2016, pp. 21–37.

[14] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.

[15] G. Jocher, A. Stoken, J. Borovec, NanoCode012, ChristopherSTAN, L. Changyu, Laughing, tkianai, A. Hogan, lorenzomammana, yxNONG, AlexWang1900, L. Diaconu, Marc, wanghaoyang0106, ml5ah, Doug, F. Ingham, Frederik, Guilhen, Hatovix, J. Poznanski, J. Fang, L. Y. , changyu98, M. Wang, N. Gupta, O. Akhtar, PetrDvoracek, and P. Rai, "ultralytics/yolov5: v3.1 - Bug Fixes and Performance Improvements," Oct. 2020. [Online]. Available: https://doi.org/10.5281/zenodo.4154370

[16] J. Courbon, Y. Mezouar, L. Eckt, and P. Martinet, "A generic fisheye camera model for robotic applications," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2007, pp. 1683–1688.

[17] S. Yogamani, C. Hughes, J. Horgan, G. Sistu, P. Varley, D. O'Dea, M. Uricár, S. Milz, M. Simon, K. Amende *et al.*, "Woodscape: A multi-task, multi-camera fisheye dataset for autonomous driving," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 9308–9318.

[18] A. Sáez, L. M. Bergasa, E. Romeral, E. López, R. Barea, and R. Sanz, "Cnn-based fisheye image real-time semantic segmentation," in *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 1039–1044.

[19] L. Deng, M. Yang, Y. Qian, C. Wang, and B. Wang, "Cnn based semantic segmentation for urban traffic scenes using fisheye camera," in *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2017, pp. 231–236.

[20] T. Li, G. Tong, H. Tang, B. Li, and B. Chen, "Fisheyedet: A self-study and contour-based object detector in fisheye images," *IEEE Access*, vol. 8, pp. 71 739–71 751, 2020.

[21] L. Deng, M. Yang, H. Li, T. Li, B. Hu, and C. Wang, "Restricted deformable convolution-based road scene semantic segmentation using surround view cameras," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 10, pp. 4350–4362, 2019.

[22] C. Häne, L. Heng, G. H. Lee, F. Fraundorfer, P. Furgale, T. Sattler, and M. Pollefeys, "3d visual perception for self-driving cars using a multi-camera system: Calibration, mapping, localization, and obstacle detection," *Image and Vision Computing*, vol. 68, pp. 14–27, 2017.

[23] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3213–3223.

[24] K. Huang, L. Wang, T. Tan, and S. Maybank, "A real-time object detecting and tracking system for outdoor night surveillance," *Pattern Recognition*, vol. 41, no. 1, pp. 432–444, 2008.

[25] A. Nazib, C.-M. Oh, and C. W. Lee, "Object detection and tracking in night time video surveillance," in *2013 10th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*. IEEE, 2013, pp. 629–632.

[26] E. Romera, L. M. Bergasa, K. Yang, J. M. Alvarez, and R. Barea, "Bridging the day and night domain gap for semantic segmentation," in *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2019, pp. 1312–1318.

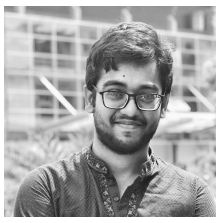
[27] D. Dai and L. Van Gool, "Dark model adaptation: Semantic image segmentation from daytime to nighttime," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 3819–3824.

[28] K. Wang and M. Z. Liu, "Object recognition at night scene based on dcgan and faster r-cnn," *IEEE Access*, vol. 8, pp. 193 168–193 182, 2020.

[29] R. Kvyetnyy, R. Maslii, V. Harmash, I. Bogach, A. Kotyra, Ż. Grądz, A. Zhanpeisova, and N. Askarova, "Object detection in images with low light condition," in *Photonics Applications in Astronomy, Communications, Industry, and High Energy Physics Experiments 2017*, vol. 10445. International Society for Optics and Photonics, 2017, p. 104450W.

[30] Y. P. Loh, X. Liang, and C. S. Chan, "Low-light image enhancement using gaussian process for features retrieval," *Signal Processing: Image Communication*, vol. 74, pp. 175–190, 2019.

Shafin Bin Hamid is currently pursuing the B.Sc. degree in Electrical and Electronic Engineering at the Bangladesh University of Engineering and Technology, Dhaka, Bangladesh. He has participated in the IEEE Signal Processing Cup 2020 and was a member of the second runner-up team 'Andromeda'. He has also participated in the IEEE Video and Image Processing Cup 2020 as a member of team 'Zodiac' and secured the second runner-up position. His current research interests include deep learning, spintronic memory devices and spintronics for neuromorphic computing.



Himaddri Roy is currently an undergraduate student conducting his B.Sc. degree in Electrical and Electronic Engineering at the Bangladesh University of Engineering and Technology, Dhaka, Bangladesh. He has been a part of the second runner-up team 'Andromeda' in the IEEE Signal Processing Cup 2020. He also represented team 'Zodiac' in the IEEE Video and Image Processing Cup 2020, securing the second runner-up position. His domain of interest includes deep learning, nanophotonics, nanoplasmonics, and plasmonic biosensors for biomolecule detection.

Munshi Sanowar Raihan is currently pursuing the B.Sc. degree in Electrical and Electronic Engineering at the Bangladesh University of Engineering and Technology, Dhaka, Bangladesh. He has participated in the IEEE Signal Processing Cup 2020 and was a member of the second runner-up team 'Andromeda'. He has also participated in the IEEE Video and Image Processing Cup 2020 as a member of team 'Zodiac' and secured the second runner-up position. His current research interests include deep learning and computer vision.

Prasun Datta is an undergraduate student of Bangladesh University of Engineering and Technology pursuing his B.Sc. degree in Electrical and Electronic Engineering. He has previously participated in the IEEE Signal Processing Cup 2020 and was a member of the team "Andromeda" which secured the position of second runner-up in the competition. He has also participated in the IEEE Video and Image Processing Cup as a team member of the team "Zodiac" securing second runner-up position. His research interests lie in signal processing, deep learning, computer vision, and biomedical signal processing.

Ashiqur Rasul is currently pursuing his B.Sc. degree in Electrical and Electronics Engineering at Bangladesh University of Engineering and Technology. He has a deep interest in signal processing and deep learning, particularly deep learning applications in material research. He participated in IEEE Video and Image Processing Cup 2020 and IEEE Signal Processing Cup 2020 and secured second runner up position in both the competitions along with his team mates. His current research interests include reinforcement learning, deep learning in material science and quantum computing.

Mohammad Ariful Haque received the B.Sc., M.Sc., and Ph.D. degrees in electrical and electronic engineering from the Bangladesh University of Engineering and Technology (BUET), Dhaka, Bangladesh, in 2003, 2005 and 2009, respectively.

He was a Postdoctoral Fellow with Concordia University, Montreal, QC, Canada. In 2003, he joined the Department of Electrical and Electronic Engineering, BUET, as a Lecturer, where he is also a Professor. He has published 37 scientific articles in internationally recognized and refereed journals and conferences. His research interests include digital signal processing, deep learning, and systems engineering.

Prof. Haque received the ReSMiQ Post-Doctoral Fellowship Award in 2012. He has supervised the champion or runner-up teams of global competitions on signal, image, and video processing organized by the IEEE Signal Processing Society at IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP) and IEEE International Conference on Image Processing (ICIP) from 2014 to 2020.