Turk. J. Math. Comput. Sci.
14(1)(2022) 174–183

DOI : 10.47000/tjmcs.1002767

Vehicle Detection from Aerial Images with Object and Motion Detection

Ibrahim Delibasoglu

Department of Software Engineering, Faculty of Computer and Information Science, Sakarya University, 54050, Sakarya, Turkey.

Received: 30-09-2021 • Accepted: 27-01-2022

Abstract. Moving vehicle detection is one of important issues in surveillance and traffic monitoring applications for aerial images. In this study, a vehicle detection method is proposed by combining motion and object detection. A method based on background modeling and subtraction is applied for motion detection, while Faster-RCNN architecture is used for object detection. Motion detection result is enhanced with the proposed superpixel based refinement method. Experimental study shows that performance of motion detection increases about 8% for F_1 metric with the proposed post processing method. Object detection, motion detection and superpixel segmentation methods interact with each other in parallel processes with the proposed software architecture, which significantly increases the working speed of the method. In last step of the proposed method, each vehicle is tracked with the kalman filter. The performance of proposed method is evaluated on the VIVID dataset. The performance evaluation shows that proposed method increases F_1 and recall values significantly compared to the motion and object detection methods alone. It also outperforms SCBU and MCD methods which are widely used for performance comparison in motion detection studies in the literature.

2010 AMS Classification: 65D19, 68U10

Keywords: Moving vehicle detection, motion detection, moving object, vehicle detection.

1. Introduction

The problem of recognizing moving objects is an important issue in areas such as real-time object tracking, event analysis and security applications. The purpose of detecting moving objects is to classify the image as foreground and background. The classification may become complex according to factors such as the motion state of the camera, ambient lighting, and dynamic changes of the background. Considering the UAV images used to monitor a region, the background is constantly changing due to camera movement. For vehicle detection from aerial images, only moving vehicles can be detected with motion information. In addition, object detection methods can be used to solve this problem, but the quality of the obtained image and the distance to the target are also extremely important for the performance of object detection methods. In cases where object detection is difficult due to image quality or distance to the target, motion detection can help to detect whether there is a moving vehicle in the image. Thus, detection performance can be increased by using object and motion detection methods together for vehicle detection.

In this study, a method that uses object and motion detection methods together is proposed. Background modelling and subtraction technique is used for motion detection. Moving vehicles are detected from aerial images by applying motion detection and Faster-RCNN is used for vehicle detection. The main contribution of this study is that the foreground mask obtained with motion detection is refined with the proposed superpixel segmentation based method. Besides, the proposed software architecture allows different processes to interact with each other by sharing data. Thus,

Email address: ibrahimdelibasoglu@sakarya.edu.tr (I. Delibasoglu)

the algorithm is divided into different parts and processed in parallel. The application has been developed using C ++, CUDA and OpenCV libraries. The remainder of this paper is organized as follows: related work is introduced in Section 2. The proposed method is explained with details in Section 3. Finally, the results are interpreted in Section 4 and conclusions are outlined in Section 5.

2. Related Work

The best results in object detection are obtained by Convolutional Neural Networks (CNN) based methods. With the increase in the number of layers to be trained in the proposed architectures, deeper neural networks were designed and the name of "deep learning" began to be used. One of the studies that played an important role in the popularity of deep learning, especially in the field of computer vision, is the AlexNet architecture. It was proposed within the ImageNet object classification competition and provided a significant performance increase [15]. With this study, the success rate in object classification has increased significantly and the performance increase in object classification has continued with different deep learning architectures such as GoogleNet [23], VGG [24] and ResNet [12]. All these proposed architectures perform feature extraction by passing the image taken as input through filters of different structures and sizes. After feature extraction, classification process is performed. The training processes of these architectures, which are trained with thousands of images, can take days due to the size of the dataset and the model complexity. Considering the success of the trained models in feature extraction, these coefficients can be used in solving different image processing problems with the method called 'transfer learning'. Thus, for a problem where we have less image set, it is possible to obtain much better results by applying transfer learning (using the trained coefficients of an existing architecture) for the first layers (feature extraction part) of the model and customizing only the last layers for our problem.

Besides architectures for image classification, different architectures have also been proposed for the problem of finding the location of an object in the image and the class to which it belongs. Unlike object classification, this problem also includes the process of finding the region that contains the object. The architectures proposed in this issue can be grouped under two main categories. In the regional-based CNN architectures proposed as the first type, there is a neural network that first selects the regions that can belong to an object, and then another neural network performs the classification process. R-CNN [11], Fast R-CNN [10] and Faster R-CNN [21] are deep learning architectures for this category. As the second type, YOLO [18] and SSD [16] architectures can be given as examples. In these network architectures, the object classification and localizing processes are done with a single neural network. Therefore, it can run faster compared to the first approach, R-CNN. In the YOLO architecture, first proposed in 2015, the input image is divided into grids and each part is responsible for whether it has an object in itself and for the class of the object. Finally, the final result is obtained by combining all the found probability values and target positions. In this context, an error function has been defined according to the estimates and ground truth objects. In addition, this proposed architecture was further enhanced over time, and versions such as yolov2 [19], yolov3 [20] and yolov4 [4] were also examined in the literature.

In motion detection applications examined in the literature; It is seen that methods based on subtraction of successive images, background modeling and optical flow are used. Although the method of basic subtraction of consecutive frames works quickly and can adapt quickly to background changes, its performance rate is very low [6, 26]. In optical flow based studies, methods that find classical dense optical flow and deep learning-based methods are used [13]. FlowNet2 architecture is one of the widely used architectures in the literature [14]. The disadvantage of deep learning methods is that the computational cost is high for specially high resolution images, so it is not always suitable for real-time applications. Therefore, it can be said that the most efficient method in terms of computation cost-performance balance is background modeling-based methods. In background modelling methods, a model of the background is constructed using certain historical frames [5]. For background modelling, classical image processing techniques [2], statistical methods [17, 28, 29] and neural networks [8] were used in different studies in the literature.

In this study, background subtraction method, which is evaluated against MCD [17] and SCBU [25] methods, proposed in [9] is used for motion detection. In MCD method, dual mode background model is constructed with mean, variance and age of each pixel. Mixing neighbouring approach is proposed to compensate the global motion to reduce the image alignment errors. In SCBU, a scene conditional background updating method is used to handle dynamic scenes issue. It tries to construct background model without contamination of the foreground. Foreground regions are extracted with foreground likelihood map by applying high and low threshold values. In method [9], neighborhood

difference is used in background subtraction and dense optical flow vectors are used as an extra information during foreground decision. Proposed method is evaluated against SCBU and MCD methods that is widely used in performance comparison for motion detection methods.

3. Methodology

3.1. **Proposed Method Architecture.** The proposed software architecture is shown in Fig. 1. The detected rectangles obtained with the object detection algorithm in process-1 are transferred to process-2 with zeroMQ and at this stage, process-1 can start to process the next frame. ZeroMQ messaging library is used to transfer detected rectangle coordinates and inform the other processes to begin to process the frame that is ready. Motion detection result is a foreground mask that can not be shared via messaging protocols in real-time. So that, shared memory is used to transfer such huge data between processes. Accordingly, the foreground mask obtained in the process-2 is shared with process-3 by using shared memory while metadata is transferred with messaging library. The foreground mask is refined by using superpixel segmentation as detailed in Section 3.4. The final results are obtained with the tracking algorithm by combining the targets found with the refined motion mask and the object detection results. Thus, a parallel working process is created in the pipeline logic.

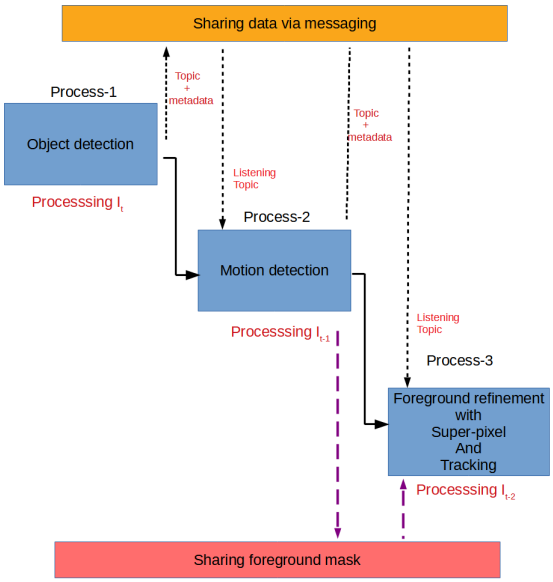


Figure 1. Proposed software architecture.

3.2. **Background Modelling and Subtraction.** In the constructed background model, the average and age information of each pixel are calculated with the following formulas (3.1) and (3.2). In the equations, i represents a pixel in the I image while $\mu_t(i)$ and $\alpha(i)$ represent the average of the corresponding pixel and the learning speed at time t, respectively;

$$\alpha(i) = \frac{1}{age(i)}, \tag{3.1}$$

$$\mu_t(i) = (1 - \alpha_t(i))\,\mu_{t-1}(i) + \alpha(i)\,I_t(i). \tag{3.2}$$

Before the background model is constructed, the displacements of the grid-based selected points on the image are estimated by the Lucas-kanade algorithm to predict motion of camera. Then, the homography matrix giving the camera movement is calculated by using the RANSAC algorithm. With the calculated homography matrix, the background model of time $t-1$ is warped to the frame at time t. After the warping process applied with the homography matrix, the age value of the pixels that newly covered in a small region by current frame is assigned to zero. The age of the

pixels in the overlapping area is increased by one. Then, mean for the background model at time t is calculated by using Eq. (3.2). Finally, moving pixels are estimated by applying subtraction between each frame and the background model with the contribution of dense optical flow as proposed in study [9].

To find bounding boxes containing the moving object, contours are extracted from the estimated moving pixel mask. At this stage, rectangles that are closer to each other more than 5 pixels are combined to fix the broken regions and a new rectangle covering the larger area is calculated. This larger rectangular region is considered as a moving object. The rectangles were accepted as correct detection in case of 20% or more intersects with the ground truth. Accordingly, the detailed performance result obtained with motion detection is shared in Table 2.

3.3. **Object Detection.** The pre-trained weights of the Faster-RCNN architecture trained on VisDrone [27] dataset captured by drone-mounted camera is used for object detection. The VisDrone dataset consists aerial images taken from 14 different cities in China. Sample frames are shown in Fig. 2 from VisDrone dataset. It has annotation for objects such as pedestrians, vehicles and bicycles. In this study, Faster-RCNN model trained on VisDrone dataset was tested on VIVID images. If the detected object rectangles overlap the ground truth by 20% or more, it is accepted as correct detection. The average performance result obtained with Faster-RCNN is shared in Table 3.

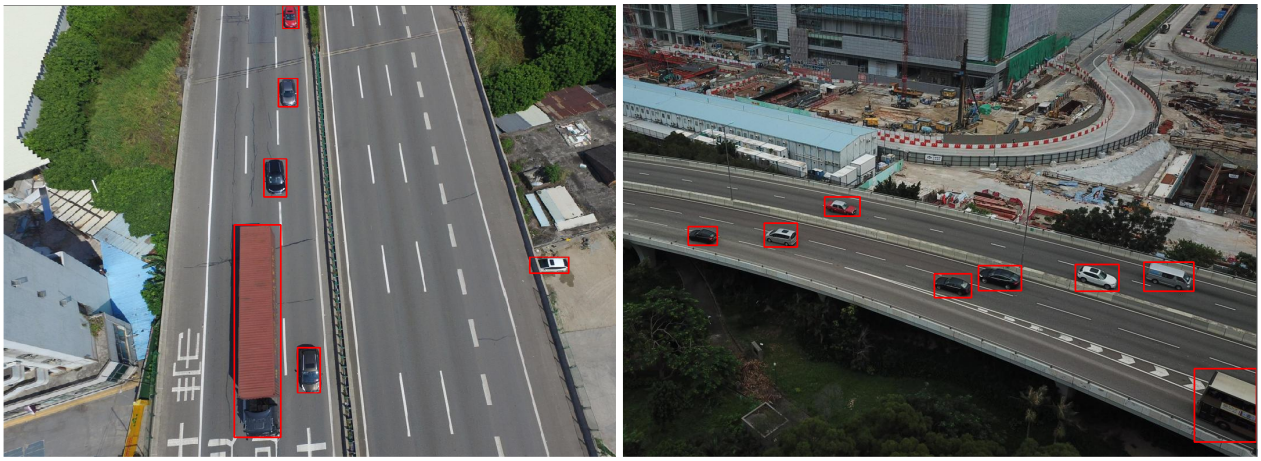


FIGURE 2. VisDrone dataset sample frames

3.4. **Foreground Refinement with Superpixel Segmentation.** Foreground mask obtained from motion detection may contain holes or disconnected regions for same object. Superpixel segmentation method is used for foreground mask refinement to enhance the motion detection result. A parallel GPU implementation of the Simple Linear Iterative Clustering (SLIC) superpixel segmentation [22] is used in this study. In the paper, it is mentioned that GPU implementation using the NVIDIA CUDA framework is up to 83x faster than the original CPU implementation of [1]. Sample frame and corresponding superpixel region borders are shown in Fig. 3

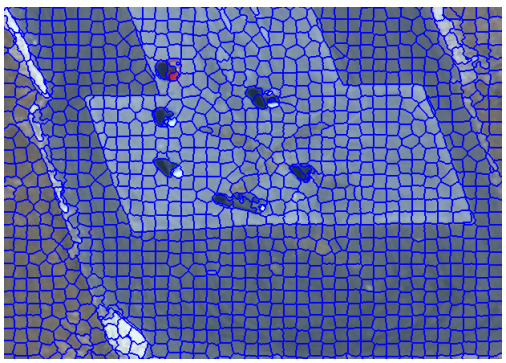


FIGURE 3. Superpixel segmentation with SLIC method

In the proposed refinement method, corresponding superpixel region is matched with the foreground mask and the ratio of foreground pixels is checked within the superpixel region. Consider that R_i^{super} and R_i^{motion} represent the corresponding regions for superpixel and motion, respectively. Ratio between R_i^{super} and R_i^{motion} intersection, and R_i^{super} is calculated for each matched superpixel region as shown in Eq. (3.3). In Eq. (3.4), Fg represents the foreground mask that is the result of motion detection and $Fg_r(R_i^{super})$ represents the refined foreground mask for the superpixel region R_i^{super}. Each superpixel region matched with foreground mask is checked, and if 20 percent of the superpixel region is detected as foreground, all pixels inside the region are set to foreground, otherwise all pixels are set to background as shown in Eq. (3.4) (T=0.2);

$$\sigma_i = \frac{R_i^{motion} \& R_i^{super}}{R_i^{super}}, \tag{3.3}$$

$$Fg_r(R_i^{super}) = \begin{cases} 0 & \sigma_i <= T \\ 255 & \sigma_i > T. \end{cases} \tag{3.4}$$

In Fig. 4, two sample regions are given with the numbers of 1 and 2. Figure 4 (a) shows the foreground mask obtained with the motion detection method introduced in Section 3.2. Corresponding superpixel region is also shown in Fig. 4 (b). Figure 4 (c) demonstrates that foreground mask is enhanced with the proposed refinement method.

Foreground mask obtained with the motion detection is compared with the refined foreground mask in experimental study to analyze the effect of using superpixel based foreground refinement method.

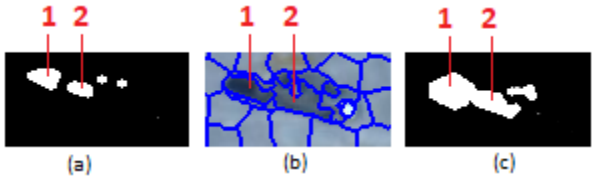


FIGURE 4. Foreground refinement with superpixel (a) foreground mask (b) superpixels (c) Refinement result

3.5. **Tracking with Object and Motion Detection.** A hybrid method combining the results of motion and object detection methods has been proposed in this study. In the proposed method, the object and motion detection algorithms run in parallel processes, and each target is tracked with the kalman tracking method after combining the results. In the tracking algorithm, the constraint of having at least three consecutive frames for each target to be valid is applied. Thus, false detections found only once in consecutive frames are prevented. Sample tracking results are shown for each sequence in Fig. 5.

4. EXPERIMENTS

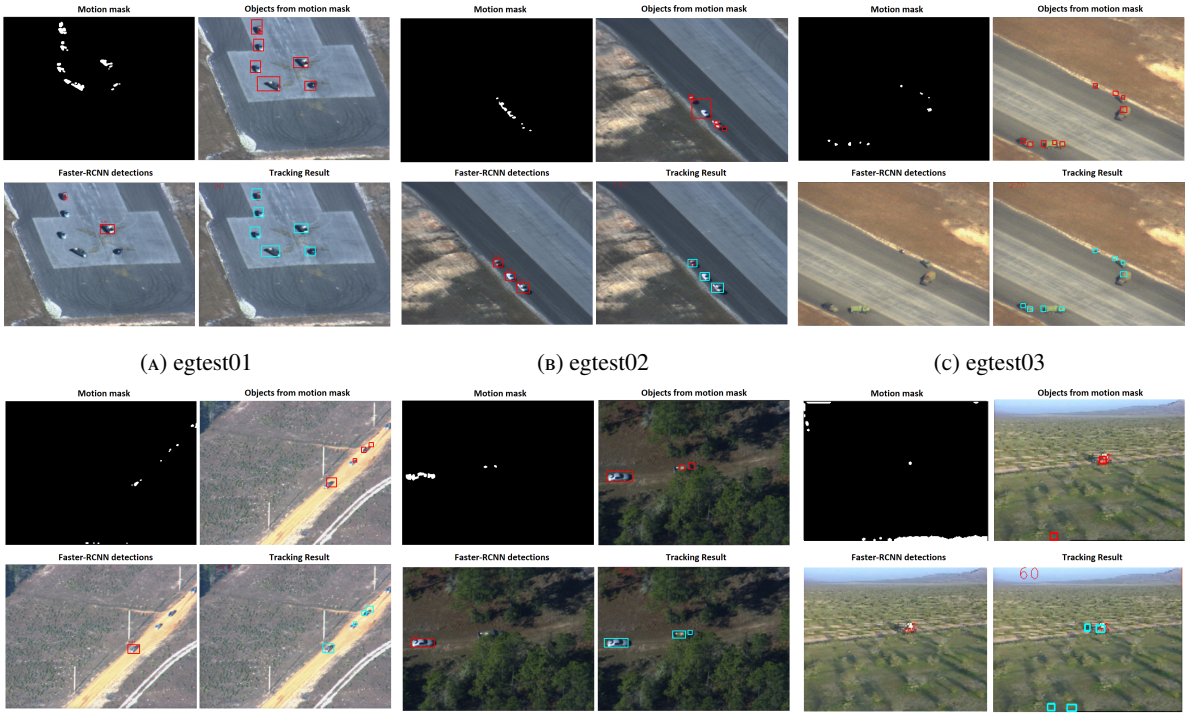
The experiments are examined on a PC with Ubuntu 18.04 operation system, AMD Ryzen 5 3600 processor with 16 GB RAM and Nvidia GeForce RTX2070 graphic card. The processing speed of each process is given in Table 1. It is seen that, motion detection and superpixel segmentation methods work much faster than object detection. The slowest process determines the overall speed of the method, since the processes work in parallel. By using proposed approach dividing the work into different processes, the method accelerates by around 45 percent. It seems possible to reach approximately 40 fps by using a faster method instead of the Faster-RCNN object detection, which is the slowest process in the pipeline.

TABLE 1. Average computation time for each process

Process	Total time
process-1 (object detection)	~80 ms
process-2 (motion detection)	~24 ms
process-3 (superpixel and tracking)	~15 ms

4.1. **Dataset.** In this study, VIVID [7] dataset consisting of aerial images of moving vehicles is used. The bounding boxes of the moving vehicles in the VIVID dataset are labeled in another dataset called LAMOD [3]. We have applied refinement to foreground mask obtained from motion detection and combined it with object detection result by tracking method. The results of motion detection detailed in Section 3.2, object detection, SCBU, MCD and proposed method are evaluated on the VIVID dataset.

4.2. **Results.** The qualitative results are shown for different sequences in Fig. 5. It is seen that in cases where Faster-RCNN could not find the vehicles for some frames, the performance can be significantly increased with motion detection. Faster-RCNN is specially unsuccessful for the vehicles in *egtest03* sequence as seen in the the figure. This may be due to the vehicle types of the images in the VisDrone dataset which used in the Faster-RCNN training. However, a major problem with motion detection is that in some cases, the entire bounding box for a moving vehicle could not be detected accurately as shown in *egtest02* and *egtest03*. Also in the experiments it is seen that, applying kalman tracking after combining detection results helps us to avoid some miss-detections.



(A) egtest01 (B) egtest02 (C) egtest03

(D) egtest04 (E) egtest05 (F) redteam

FIGURE 5. Obtained foreground mask and Faster-RCNN detections for example frames in different sequences

The quantitative performance of the methods is measured with Precision, Recall and F-score(F_1) which are defined in Eq. (4.1) and (4.2). In these equations; FP refers wrong detections, TP refers true detections and FN refers the missed vehicles;

$$Precision = \frac{TP}{TP + FP}, \quad Recall = \frac{TP}{TP + FN}, \tag{4.1}$$

$$F_1 = \frac{2 * Precision * Recall}{Precision + Recall}. \tag{4.2}$$

The motion detection result is compared with the refinement result to analyze effect of using superpixel based foreground refinement method. In Table 2, detection results from foreground and refinement masks are compared with the performance criteria: precision, recall and F_1. It is seen that proposed refinement method increases for all three

metrics for all sequences. Average F_1 value with refinement mask increases from 0.5553 to 0.6067, precision from 0.5421 to 0.6183 and recall from 0.5879 to 0.6147.

TABLE 2. Performance comparison of motion and superpixel refinement results

VIVID sequence		***Motion***	***Proposed (Motion + superpixel)***
egtest01	*Precision*	0.8291	**0.8755**
	Recall	0.8844	**0.8939**
	F_1	0.8559	**0.8846**
egtest02	*Precision*	0.6669	**0.7407**
	Recall	*0.5283*	**0.5500**
	F_1	*0.5896*	**0.6313**
egtest03	*Precision*	0.3078	**0.4731**
	Recall	0.2158	**0.2993**
	F_1	0.2537	**0.3667**
egtest04	*Precision*	0.6377	**0.6871**
	Recall	**0.6806**	0.6780
	F_1	0.6584	**0.6825**
egtest05	*Precision*	0.4394	**0.5205**
	Recall	0.6517	**0.6968**
	F_1	0.5249	**0.5959**
redteam	*Precision*	0.3720	**0.4134**
	Recall	0.5671	**0.5705**
	F_1	0.4493	**0.4794**

It is important that morphological opening is applied for all motion detection methods to remove the noise. Also, detected bounding boxes within 5 pixels from the borders are ignored due to much wrong detections caused by camera motion in these regions.

In Table 3, average metric values are given for each method. Faster-RCNN method has best average precision value, but it has also the lowest recall value. By combining motion and object detection method, proposed method achieves best average recall and F_1 values with 0.7658 and 0.6943, respectively. MCD has similar performance with our proposed motion with superpixel method. Even if SCBU outperforms MCD method as shown in SCBU paper for a different dataset, we obtain poor performance with SCBU in this study. SCBU has lowest average F_1 after Faster-RCNN. The quantitative performance comparison for each sequence is shown in Fig. 6, 7 and 8 for each metric. In Fig. 7, it seems that Faster-RCNN has the lowest recall performance for most sequences. The performance of the proposed method is particularly high for the *egtest02*, *egtest05* and *redteam* sequences in recall metric. Faster-RCNN has best precision in most of the sequences except *redteam*. In Fig. 8, it seems that proposed method increases the F_1 value for each sequence except *egtest05* against motion and Faster-RCNN. Only, MCD method has better performance against proposed method for F_1 metric in *redteam* sequence. In brief, it is clear that the performance of motion and object detection methods has been increased with the proposed approach and proposed method outperforms MCD and SCBU methods in overall.

TABLE 3. Average performance of each method

Method	**Precision**	**Recall**	F_1
Motion	0.5421	0.5879	0.5553
Motion with superpixel	0.6183	0.6147	0.6067
Faster-RCNN	**0.8672**	0.4042	0.4856
MCD	0.6495	0.5967	0.6036
SCBU	0.5361	0.4870	0.5048
Proposed	0.6722	**0.7658**	**0.6943**

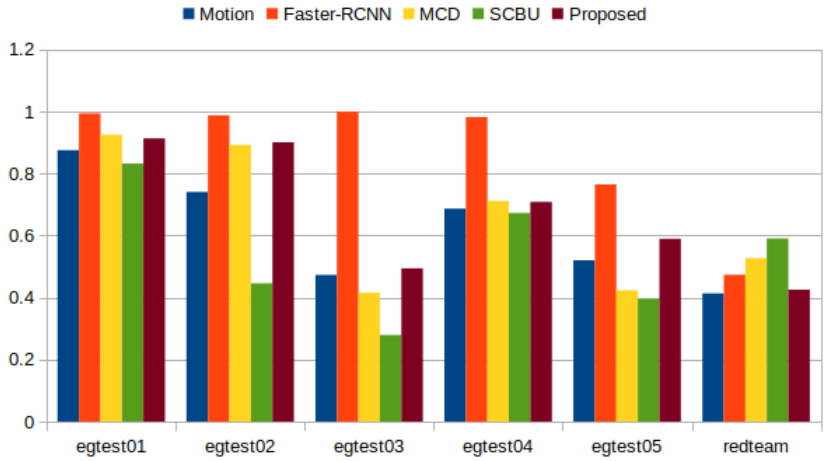


FIGURE 6. Precision comparison of detection results

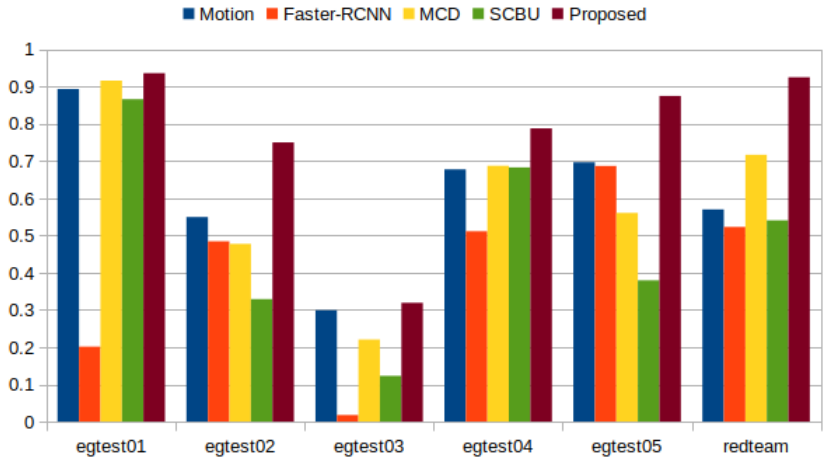


FIGURE 7. Recall comparison of detection results

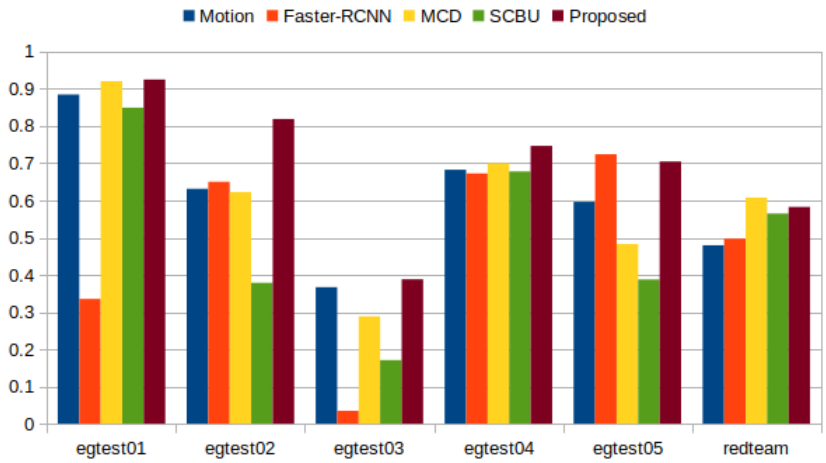


FIGURE 8. F_1 comparison of detection results

5. Conclusions

In this paper, a software architecture is proposed to be able to run object detection, motion detection and superpixel algorithms in parallel processes. Motion detection is enhanced with proposed superpixel based refinement method. Kalman tracking is applied after combining motion and object detection results. The experimental results show that motion detection method has better recall value compared to object detection for aerial images. Proposed method increases recall and F_1 values compared to motion and object detection methods alone. While average F_1 value for proposed method is 0.6943, motion and object detection methods have average 0.5553 and 0.4856 F_1 values, respectively. We have also evaluated the effectiveness of the proposed method by comparing it with state-of-the-art methods SCBU and MCD. Experimental study shows that proposed method has robust performance against MCD and SCBU. When the proposed method is examined in terms of speed, it is observed that motion detection method works faster compared to the Faster-RCNN. Different architectures can be tested for object detection to be able to increase the accuracy and speed of the proposed method, as the slowest process determines the overall speed of the method.

Conflicts of Interest

The author declares that there are no conflicts of interest regarding the publication of this article.

Authors Contribution Statement

The author has read and agreed to the published version of the manuscript.

References

[1] Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P. et al., *SLIC superpixels compared to state-of-the-art superpixel methods*, IEEE Transactions On Pattern Analysis And Machine Intelligence, **34**(2012), 2274–2282.

[2] Allebosch, G., Deboeverie, F., Veelaert, P., Philips, W., *EFIC: edge based foreground background segmentation and interior classification for dynamic camera viewpoints*, International Conference On Advanced Concepts For Intelligent Vision Systems, (2015), 130–141.

[3] Berker Logoglu, K., Lezki, H., Kerim Yucel, M., Ozturk, A., Kucukkomurler, A. et al., *Feature-based efficient moving object detection for low-altitude aerial platforms*, Proceedings Of The IEEE International Conference On Computer Vision Workshops, (2017), 2119–2128.

[4] Bochkovskiy, A., Wang, C., Liao, H., *Yolov4: Optimal speed and accuracy of object detection*, ArXiv Preprint ArXiv:2004.10934, (2020).

[5] Bouwmans, T., Hofer-lin, B., Porikli, F., Vacavant, A., Traditional Approaches in Background Modeling for Video Surveillance, Handbook Background Modeling And Foreground Detection For Video Surveillance, Taylor And Francis Group, 2014.

[6] Collins, R., Lipton, A., Kanade, T., Fujiyoshi, H., Duggins, D. et al., *Others A system for video surveillance and monitoring*, VSAM Final Report, (2000), 1.

[7] Collins, R., Zhou, X., Teh, S., *An open source tracking testbed and evaluation web site*, IEEE International Workshop On Performance Evaluation Of Tracking And Surveillance, **2**(2005), 35.

[8] De Gregorio, M., Giordano, M., WiSARDrp for Change Detection in Video Sequences, ESANN, 2017.

[9] Delibasoglu, I., *UAV images dataset for moving object detection from moving cameras*, ArXiv E-prints, earXiv:2103.11460, (2021).

[10] Girshick, R., *Fast r-cnn*, Proceedings Of The IEEE International Conference On Computer Vision, (2015), 1440–1448.

[11] Girshick, R., Donahue, J., Darrell, T., Malik, J., *Rich feature hierarchies for accurate object detection and semantic segmentation*, Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition, (2014), 580–587.

[12] He, K., Zhang, X., Ren, S., Sun, J., *Deep residual learning for image recognition*, Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition. (2016), 770–778.

[13] Huang, J., Zou, W., Zhu, J., Zhu, Z., *Optical flow based real-time moving object detection in unconstrained scenes*, ArXiv Preprint ArXiv:1807.04890, (2018).

[14] Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy et al., *Flownet 2.0: Evolution of optical flow estimation with deep networks*, Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition, (2017), 2462–2470.

[15] Krizhevsky, A., Sutskever, I., Hinton, G., *Imagenet classification with deep convolutional neural networks*, Advances In Neural Information Processing Systems, **25**(2012), 1097–1105.

[16] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S. et al., *Ssd: Single shot multibox detector*, European Conference On Computer Vision, (2016), 21–37.

[17] Moo Yi, K., Yun, K., Wan Kim, S., Jin Chang, H., Young Choi, J., *Detection of moving objects with non-stationary cameras in 5.8 ms: Bringing motion detection to your mobile device*, Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition Workshops, (2013), 27–34.

[18] Redmon, J., Divvala, S., Girshick, R., Farhadi, A., *You only look once: Unified, real-time object detection*, Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition, (2016), 779–788.

[19] Redmon, J., Farhadi, A., *YOLO9000: better, faster, stronger*, Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition, (2017), 7263–7271.

[20] Redmon, J., Farhadi, A., *Yolov3: An incremental improvement*, ArXiv Preprint ArXiv:1804.02767, (2018).

[21] Ren, S., He, K., Girshick, R., Sun, J., *Faster r-cnn: Towards real-time object detection with region proposal networks*, ArXiv Preprint ArXiv:1506.01497, (2015).
[22] Ren, C., Prisacariu, V., Reid, I., *gSLICr: SLIC superpixels at over 250Hz*, ArXiv E-prints, (2015).
[23] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S. et al., *Going deeper with convolutions*, Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition, (2015), 1–9.
[24] Simonyan, K., Zisserman, A., *Very deep convolutional networks for large-scale image recognition*, ArXiv Preprint ArXiv:1409.1556, (2014).
[25] Yun, K., Lim, J., Choi, J., *Scene conditional background update for moving object detection in a moving camera*, Pattern Recognition Letters, **88**(2017), 57–63.
[26] Zhao, L., Tong, Q., Wang, H., *Study on moving-object-detection arithmetic based on W4 theory*, 2011 2nd International Conference On Artificial Intelligence, Management Science And Electronic Commerce (AIMSEC), (2011), 4387–4390.
[27] Zhu, P., Wen, L., Bian, X., Ling, H., Hu, Q., *Vision meets drones: A challenge*, ArXiv Preprint ArXiv:1804.07437, (2018).
[28] Zivkovic, Z., *Improved adaptive Gaussian mixture model for background subtraction*, Proceedings Of The 17th International Conference On Pattern Recognition, ICPR 2004, **2**(2004), 28–31.
[29] Zivkovic, Z., Van Der Heijden, F., *Efficient adaptive density estimation per image pixel for the task of background subtraction*, Pattern Recognition Letters, **27**(2006), 773–780.