IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Traitement du Signal
Vol. 37, No. 2, April, 2020, pp. 209-216
Journal homepage: http://iieta.org/journals/ts

Hand-Crafted Features vs Deep Learning for Pedestrian Detection in Moving Camera

Bilel Tarchoun[1], Anouar Ben Khalifa[1*], Selma Dhifallah[1], Imen Jegham[2], Mohamed Ali Mahjoub[1]

[1] Université de Sousse, Ecole Nationale d'Ingénieurs de Sousse, LATIS-Laboratory of Advanced Technology and Intelligent Systems, Sousse 4023, Tunisia
[2] Université de Sousse, Institut Supérieur d'Informatique et des Techniques de Communication de H. Sousse, LATIS-Laboratory of Advanced Technology and Intelligent Systems, Sousse 4011, Tunisia

Corresponding Author Email: anouar.benkhalifa@eniso.rnu.tn

https://doi.org/10.18280/ts.370206

Received: 7 January 2020
Accepted: 8 March 2020

Keywords:
deep learning, handcrafted features, intelligent transport systems, moving camera, pedestrian detection

ABSTRACT

Detecting pedestrians and other objects in images taken from moving platforms is an essential task needed for many applications such as smart surveillance systems and intelligent transportation systems. However, most detectors in this domain still rely on handcrafted features to separate the foreground objects from the background. While these types of methods have presented good results with good response times, they still have some weaknesses to overcome. In recent years, alternative object detection methods are being proposed, with deep learning based approaches rising in popularity thanks to their promising results. In this paper, we propose two pedestrian detectors for use in images taken from a moving vehicle: The first detector uses a block matching algorithm and handcraft features for pedestrian detection, and the second uses a Faster R-CNN deep detector. We also compare both systems' performances to other state-of-the-art pedestrian detectors. Our results show that although handcraft feature-based approach achieves good results within acceptable detection times, it suffers from a high false positive rate. However, we found that Faster R-CNN detector performs better in terms of precision and recall, but these improved results come at a cost of detection time.

1. INTRODUCTION

The development of autonomous vehicles is attracting the interest of the computer vision community since these vehicles require an array of sensors such as radar, lidar, cameras and ultrasonic sensors to perceive their environment [1, 2]. Among these sensors, cameras are the most popular tool for intelligent transportation systems as they provide an extensive amount of information [3, 4]. This information is used to detect objects present around the vehicle. Pedestrian detection is the most critical part of these needed detection tasks to ensure the safety of all of the vehicles and people on the road. The need for such pedestrian detectors is an important driving force behind the evolution of machine learning and computer vision, with many detection methods proposed each year [5, 6]. However, most of these proposals rely on the assumption that the camera used for the detection task is static. These methods fail to produce the required accuracy when dealing with moving targets in a moving environment, which is the case for autonomous vehicles [7]. These issues have created a demand for pedestrian detection approaches that are able to adapt to the autonomous vehicle needs. But this further complicates an already challenging task. In addition to the challenges present in pedestrian detection from a static camera such as pedestrian appearance variations, non-rigid deformations, illumination problems and occlusions, the detection becomes more complex with the camera's motion [8, 9]. This complexity is caused by the difficulty in distinguishing the foreground objects from the background, as the background itself is also moving.

To overcome these challenges, many methods for detecting objects and separating them from their backgrounds have been proposed. These methods can be categorized in two different families: Methods based on hand crafted features that model the background using techniques such as Aggregated Channel Features, Gaussian Mixture Models and Optical flows [10-12]. The second family uses the recent developments in deep learning that were made possible in the last decade thanks to the availability of larger training databases and more powerful hardware [13, 14].

In this paper we propose two pedestrian detectors, the first detection framework is based on a block matching algorithm for camera motion compensation, and handcraft features for object classification. The second detector is based on a deep neural network framework to detect pedestrians by a camera mounted in the dashboard of a car. Our deep neural network is based on a Faster R-CNN [15] architecture trained using a transfer learning scheme. We compare our detectors performances to each other and several handcrafted feature based detectors. Our Faster R-CNN detector outperforms hand craft feature based detectors thanks to the ability of CNNs to generate discriminant features even in real-world conditions. The main contributions of this paper are the following:

- We propose a handcraft feature based detector that uses a block matching algorithm to compensate the camera movement.
- We propose a deep pedestrian detector designed for use with a camera installed on a vehicle.
- We compare the performance of our detector to other handcraft feature based detectors.

The rest of the paper is organized as follows. Section 2 introduces a survey on recent methods for detecting moving objects by using a moving camera. Section 3 presents both of our proposed detectors for use in moving cars. In section 4, we evaluate our detectors performances and compare the results to each other and to other pedestrian detectors. And finally, the last section concludes the paper.

2. RELATED WORK ON OBJECT DETECTION WITH A MOVING CAMERA

Pedestrian detection is a widely studied field in computer vision, with many feature extraction and classification methods already proposed in the literature. The majority of these works are concerned with the case of pedestrian detection in static cameras. In our work, we focus on the context of a camera mounted on a moving platform. Object detection with a moving camera introduces additional issues related to camera motion, and classic methods of background modelisation and background subtraction approaches cannot be used in this case. There are few works that address the problem of pedestrian detection on a moving platform. These methods can be divided into two categories: Methods that use handcrafted features, and methods that adopt a deep learning based detector.

2.1 Methods based on handcrafted features

These methods are based on the separation of the foreground of the foreground objects from the images background, then extracting features from the segmented objects and classifying them.

Cho et al. [10] implemented a moving object detector for advanced driver assistance systems based on both optical flow and background subtraction techniques to mitigate false positives. First, optical flow is estimated from the current and previous frame and used to estimate the camera's egomotion. The motion information is then used to apply motion compensation on the previous frame and to generate the background model using a Gaussian Mixture Model. Using this information, the authors applied two types of detection to the image by subtracting the background model from the current frame and by detecting the difference in optical flows between the current frame and the previous frame with motion compensation applied. Finally, the results from both detection techniques were cross-checked to validate the detections and remove false positives. The authors have also successfully implemented their detection method on an FPGA board and detailed the hardware structure of their approach.

Zhang et al. [11] present an object detector for images taken from a camera mounted on an unmanned aerial vehicle. The authors proposed an optical flow based solution for the detection task, but since the UAV moves in a 3D space the optical flow vectors could not be directly compared. To solve this issue, the authors proposed to create a homography between the previous frame and the current frame to adjust the optical flow vectors. With these adjustments, it is possible to compare the difference between the optical flow vectors and the background movement to generate candidate detections. Next, the authors proposed to refine the detections by clustering close regions and using convex hull filling to remove the cracks in the candidates. Finally, the authors tested for false positives by removing candidates that have abrupt motions between frames, as an object should only move for a small distance in a short time.

Yun et al. [16] presented a method to detect moving objects in a moving camera view based on a background subtraction method. The detection includes 3 steps, in the first step various variables are extracted from the scene such as the illumination variation, background motion extracted with the Lucas-Kanade optical flow algorithm in the form of a homography between the current and previous frame, and a foreground motion estimation based on the difference between the current frame and the previous frame adjusted by the background motion homography. In the next step, the background model is updated using the homography calculated in the previous step and the illumination change. Finally, each pixel's probability to be in the background is calculated based on the difference between the pixel's intensity and the background model data relevant to the pixel, and these pixel probabilities are thresholded to obtain the foreground elements after solving ambiguities according to their connectivity to known foreground elements.

Yu et al. [17] propose to detect moving objects in a moving camera view by aligning each frame with the previous one using optical flows. To find corresponding points between frames, the Lucas-Kanade optical flow algorithm is applied on a grid of 16 × 16 evenly distributed grid of points on the previous frame. Then, each point in the grid is paired up with their resulting point in the current frame. Six of these corresponding points are chosen according to the RANSAC algorithm in order to calculate a homography matrix which is then used to align the pair of frames. A background model is then updated using the difference between the current frame and the previous model's iteration, as well as the homography. To find the moving objects, the current frame is aligned with the two previous frames and compared to the updated background model to produce a binary image containing the moving objects. Finally, the objects in this image are segmented using filters and morphological operations.

All the presented methods included 2 common steps which are the creation of a background model and the compensation of the camera motion. These steps are important in order to detect elements deviated from this model as objects of interest. While the methods used to create these models (most commonly optical flows and Gaussian mixture models) produce good results, there are still some noticeable issues with the results such as noisy detections and difficulties in detecting very small or very large objects.

2.2 Methods based on deep learning

This family of methods relies on the ability of CNNs to generate discriminative features for the detection task without necessarily needing a background detection technique or other camera pre-processing.

Heo et al. [18] introduced a deep learning approach for detecting objects in a dynamic background. This approach relies on the fusion of two CNNs for the detection task, the first CNN detects the appearances of objects using the current frame as an input while the second CNN detects motion information using the current frame and the background model. For the first network's architecture, the authors fine-tuned the layers preceding the pool4 layer the VGG-16 network and added a convolution layer that performs the appearance based detection. While for the second network, the authors trained a shallow network composed of three convolutional layers and

a pooling layer as higher level data is irrelevant for detecting motion information. The two networks are then fused into one using a pixel wise SoftMax layer and re-trained to obtain the final detector.

To improve detection results for small size pedestrians and better distinguish them from background elements, Kong et al. [13] introduced a modified Faster R-CNN architecture that combines multi-scale features and contextual information. In their work, the authors asserted that the shortcomings of Faster R-CNN detectors in detecting smaller pedestrians and discriminating them from the background are caused by the low resolution of features extracted from small scale pedestrians in the final convolution layers as well as the ROIs for feature extraction being too close to the detection target. To solve this issue, the authors proposed to pool features from a region 1.5× larger in size than the original ROI proposal and also to pool features from multiple convolution layers with an additional normalization and scaling step to obtain a discriminative feature set better suited for detecting difficult targets.

In order to detect pedestrians for a human aware navigation application for robots, Mateus et al. [14] proposed to combine a hand-crafted detector with a convolutional neural network detector to achieve good detection performances while keeping an acceptable detection speed. Instead of processing the whole image with a CNN detector, the authors first generated pedestrian proposals using an Aggregated Channel Features (ACF) pedestrian detector. These proposals are further filtered by removing proposals that have a detection score below a certain threshold. For the next step, the authors have chosen to implement a VGG-VD16 model that was fine-tuned on the INRIA dataset and modified to fit the application's needs. This detector is then used to further classify the ACF detector's proposals in order to either accept them for the final detection results or otherwise reject them.

Rozantsev et al. [19] proposed two approaches to detect unmanned aerial vehicles (UAV) and aircraft in images taken by a mobile camera. This task presents additional challenges compared to detecting objects on the ground since UAVs and aircraft move in a 3D space, and the background can be extremely complex (changing between sky and ground). In both approaches, the video is divided in overlapping slices of frames and these slices are further divided by a sliding window to form "spatio-temporal cubes" (st-cubes). Then motion compensation is applied to stabilize the st-cubes, and these st-cubes are classified depending on the presence of an object. The two approaches differ in the methods used for motion compensation and the classification step. In the first approach, the authors use two regressors based on boosted trees for motion compensation in the x and y direction and classification is done by using HOG3D features and the gradient boost algorithm. The second approach is based on deep learning techniques, two CNNs are trained for the motion compensation step to first coarsely align the large movements and then refine the small changes. Another CNN is trained for the classification step to detect the presence of objects in the motion compensated st-cubes. The authors conclude that the CNN based approach has led to the best detection results.

CNN based detection is increasingly being adopted for many applications, and the task of detecting moving objects in a moving camera is no exception. These deep learning based approaches have shown great results compared to their handcraft feature counterparts thanks to their ability to learn task specific features without necessarily needing pre-processing such as motion compensation. However, CNN based methods are still much slower than real-time processing requirements, and the need of powerful computation hardware make these solutions impractical in certain scenarios.

3. PROPOSED METHODS

In this section we detail our proposed frameworks for pedestrian detection by a moving camera. The first framework is based on deep neural network detection. The second framework is based on handcrafted features for pedestrian detection by compensating the camera's motion.

3.1 Deep learning detector

We have chosen a deep learning based approach for the detection task as convolutional neural networks have proven their ability to learn robust and distinguishing features for the detection targets even in difficult situations. Our method will be discussed in two parts as shown in Figures 1 and 2. In the first part, we prepare our detector using transfer learning, while in second part we apply our trained detector in order to detect pedestrians in images captured by the vehicle's camera.

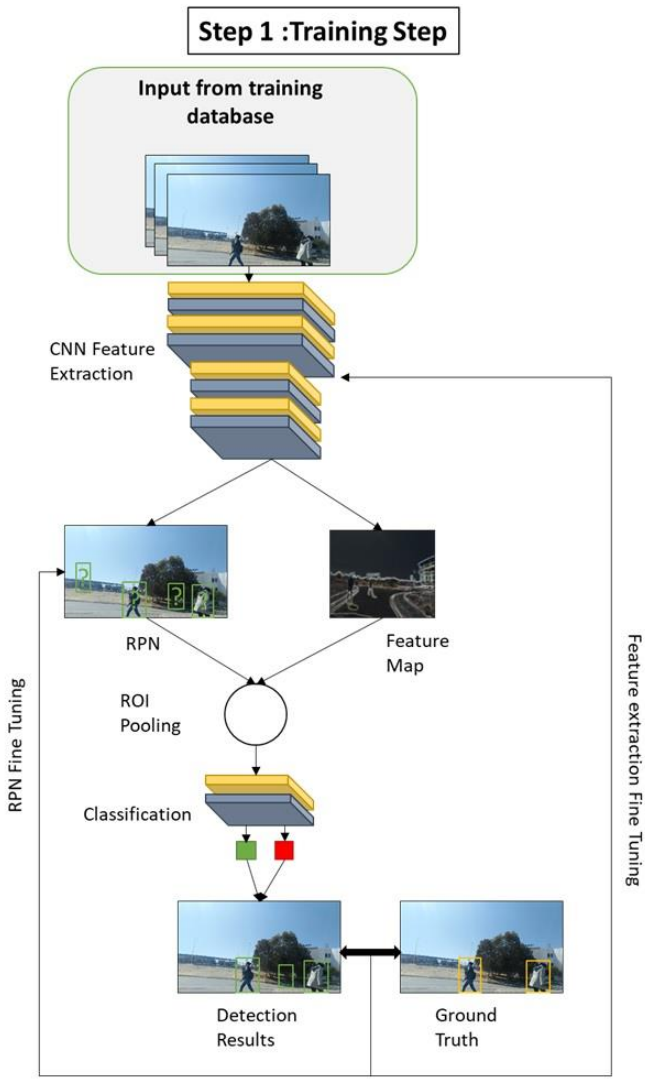


Figure 1. Deep detector training step

Step 1: Detector training

Our detector uses a Faster R-CNN based architecture as its basis, as Faster R-CNN based architectures have shown the best speed to performance compromise compared to other CNN architectures [20]. For training the proposed detector we choose to use transfer learning techniques since they allow us to retain the advantages of CNN models trained on large databases. Transfer learning also allows us to fine-tune the detector using less training data and requiring much less training time to reach optimal results. To obtain our training data, we divided our database in two sets, one set is used for training (1) and the other one is used for validation (2).

$$DB_{train} = \{(I, G)_k\}_{k=1}^{|DB_{train}|} \quad (1)$$

$$DB_{test} = \{(I, G)_k\}_{k=1}^{|DB_{test}|} \quad (2)$$

where, I_k represents the image number k in the dataset and g_k represents the associated set of ground truth bounding boxes, with $g_k = [x_k^g, y_k^g, w_k^g, h_k^g]$ being the x,y coordinates of the top left corner, the width and the height of the bounding box.

The training database is then used for fine tuning a Faster R-CNN in a four step process where the weights of the Region Proposal Network (RPN) and the CNN feature extraction layers are alternatively adjusted: First, the RPN is re-trained while the CNN weights are locked, then then feature extraction layers are re-trained. These two steps are repeated again to fine-tune the training.

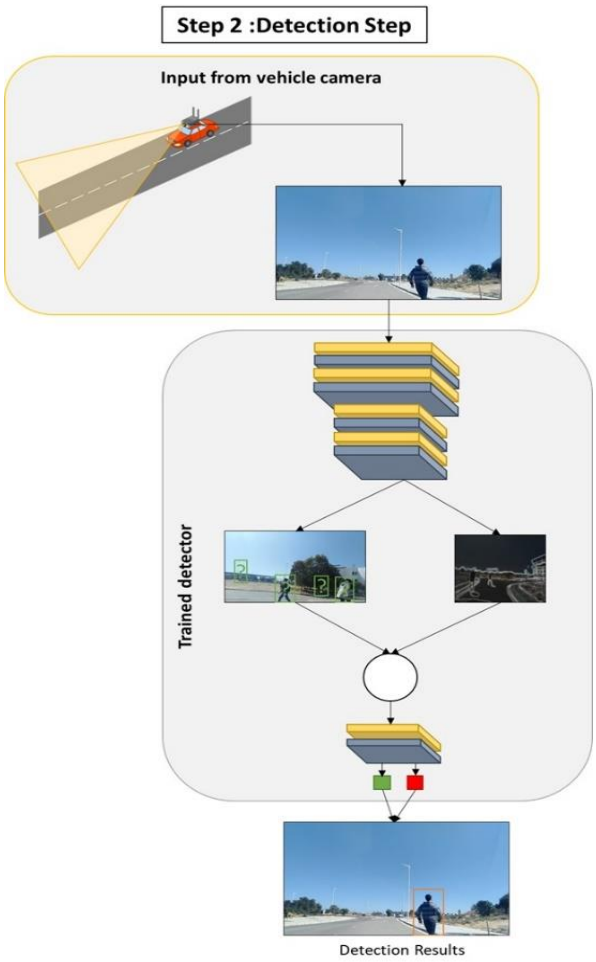


Figure 2. Pedestrian detection in vehicle mounted camera images using our deep detector

Step 2: Detection step

At a given time t, the vehicle mounted camera will capture a video frame It which is subsequently sent for processing by our proposed detector for the pedestrian detection task. Object detection in Faster R-CNN architectures consists of two tasks.

Multiscale feature maps are extracted from the input image by the convolution and pooling layers that form the feature extraction part of the network. Also, the RPN suggests potential areas where objects may be present, these areas are suggested around anchor points in different sizes and aspect ratios. The feature maps and the areas suggested by the RPN are then pooled together in the ROI pooling step, and the final classification layers decide if a certain area contains a pedestrian or not. The output of our detector is a set of pedestrian bounding boxes $BBox_t = \{[x_i, y_i, w_i, h_i]\}_{i=1}^{|BBox_t|}$.

3.2 Handcraft features detector

The main idea of this method consists of the fact that, in object detection using a moving camera, the foreground movement is different from the background motion. The background's motion is constrained by the camera's limited range of movement, while foreground objects can move freely in any direction. Our proposed handcraft features based approach is shown in Figure 3. The main steps of our approach are:

Step 1: Region of interest extraction

Given two consecutive images, we extract a region of interest (ROI) that is unlikely to contain foreground objects. In our case, we choose the top 20% of the image as our ROI, as in a driving scenario foreground objects are mostly situated on the ground level, and this region still contains enough background information to properly estimate background motion

Step 2: Background motion estimation

In this step, we estimate the background's motion with a block matching algorithm [7], we split our selected ROIs in blocks of equal size. For each block in frame t, we search the corresponding block in frame t + 1. Different block matching algorithms have been evaluated to calculate the global motion vector $\vec{V_t}$ at time t as shown by (3).

$$\overrightarrow{V_t} = \frac{b^2}{P \times Q} \sum_{l=1}^{\frac{P}{b}} \sum_{s=1}^{\frac{Q}{b}} \overrightarrow{V_{l,s}} \quad (3)$$

With:

$b \times b$: Size of the different blocks; $P \times Q$: Size of each ROI image; $\overrightarrow{V_{l,s}}$: Motion vector between two consecutive images for block (l,s).

Step 3: Foreground object detection

In this step we divide the rest of the frames into blocks, and match each block in frame t with its corresponding block in frame t+1 using the same block matching algorithm chosen for the previous step to calculate a movement vector for each block. The obtained movement vectors are then compared to the background movement vector using the Euclidean distance. If the difference between the movement vectors is greater than a certain threshold, the block is considered as a foreground element and assigned a value of 1, otherwise the block is considered as a part of the background and assigned the value of 0. These values are then used to create a binary image that shows the foreground objects.

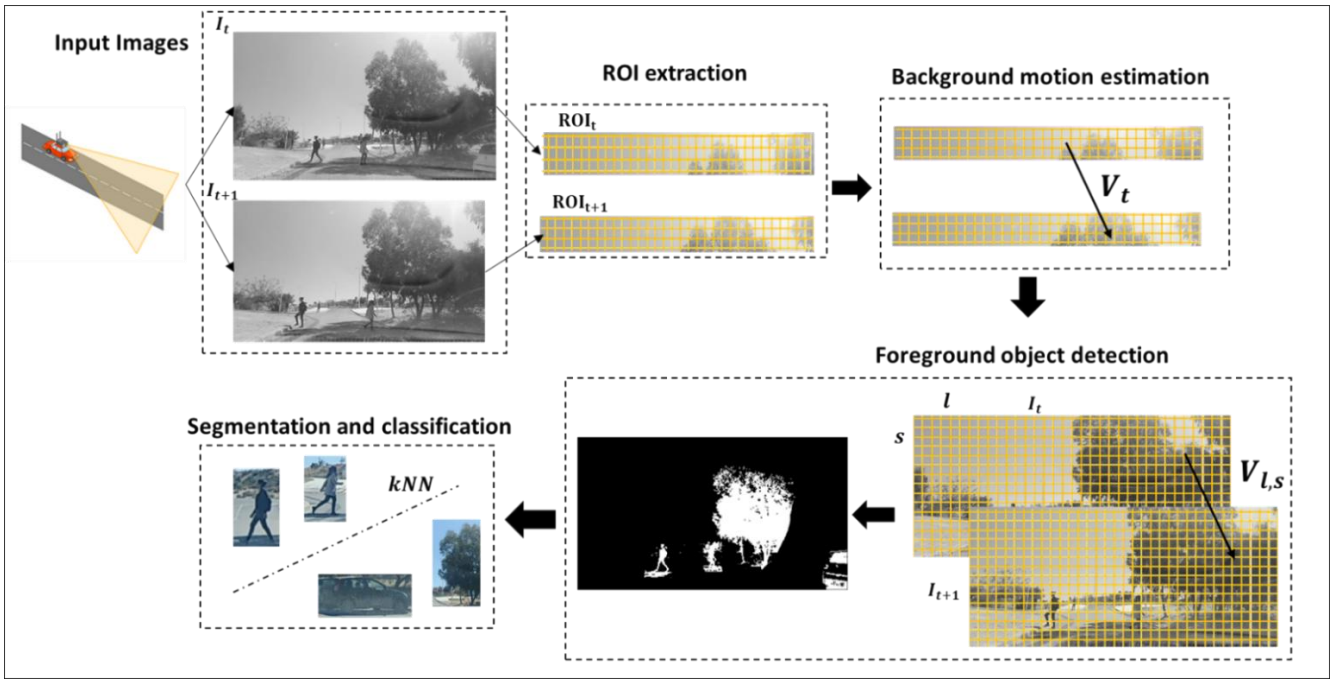


Figure 3. Schematic of our proposed handcraft feature based detector

Step 4: Segmentation and classification

We first separate the different foreground objects found in the binary image using the method described by Haralick and Shapiro [21]. Then we extract feature vectors from each segmented object using the Histogram of Oriented Gradients (HOG) method [22]. The final task is to use the extracted feature vectors in order to classify each object in a pair of classes: Pedestrian and Non-Pedestrian. We opted for a kNN based classifier due to their high accuracy while keeping computational times low.

4. EXPERIMENTS AND RESULTS

In this section, we evaluate our proposed pedestrian detectors and compare their performances to each other and other handcraft feature based detectors [23] using the vehicle half of the I2V-MVPD database [24]. The database contains a total of sixty-one video sequences of various scenarios in which a vehicle moves while pedestrians cross the road, these sequences contain a total of 4740 images. As discussed in the previous section, we divide the database in two sets, the first twenty sequences are used as training data, while the rest of the sequences will form the testing data. All of the experiments done in this section have been performed on a computer with the following specifications: CPU: Intel Core i7-7700HQ, RAM: 16 GB, GPU: NVIDIA GTX 1050Ti with 4GB of VRAM.

4.1 Deep learning detector training parameters

We considered five different pre-trained CNN architectures to fine-tune for our detector: VGG-16, VGG-19, Resnet-18, Mobilenetv2 and GoogleNet. These models were pre-trained using the ImageNet database, which is a very large database containing one million images divided into one thousand classes of various objects. But before proceeding to our finetuning step, we need to slightly modify these networks. Since these networks were pre-trained using one thousand classes, their classification layers contain one thousand outputs. However, we only need two outputs: the presence or absence of a pedestrians. Therefore, we keep only two outputs and change the connections on the previous layers to match our new output layer. For the fine-tuning, we use the hyperparameters shown in Table 1.

Table 1. Hyperparameters for the training step

Hyperparameter	Value
Mini Batch Size	4
Learning Rate	10^{-4}
Number of epochs	20
Number of strongest regions	500
Momentum	0.9

4.2 Results

We evaluated the results of our five trained Faster R-CNN architectures and our handcraft features based detector as well as an Aggregate Channel Features (ACF) [25] based detector [12] and the method described in [23]. The evaluation metrics we used are the following:

- Precision (P): The percentage of correct detections among all detections.
- Recall (R): The percentage of correct detections among all objects to detect.
- Average Precision (AP): The surface under the precision/recall curve.
- Detection Time (T).

The detection results are shown in Table 2.

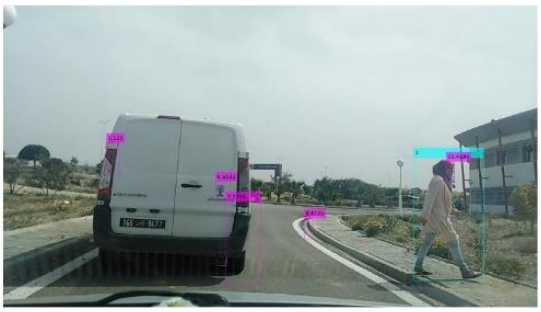
The results show that our proposed handcraft performs better than the two other tested handcraft detectors. This performance is especially noticeable in the precision metric where our detector significantly outperforms the other two detectors. But our detector has a slightly slower detection time compared to the other detectors, with an additional 50ms of detection time per frame.

Table 2. Detection results

	Architecture	AP (%)	P (%)	R (%)	T (s)
Handcraft features	ACF [25]	52.06	14.87	71.55	0.214
	SURF interest points [20]	53.14	16.65	71.45	0.225
	Proposed	**64.78**	**43.15**	**76.06**	**0.270**
Deep Learning	Proposed VGG-16	52.59	40.34	60.00	0.503
	Proposed VGG-19	57.41	61.81	64.01	0.580
	Proposed Googlenet	61.75	66.23	65.21	0.628
	Proposed Mobilenetv2	64.99	43.28	76.15	0.619
	Proposed Resnet-18	**71.82**	**57.32**	**78.18**	**0.584**

Our second proposed detector based on a deep learning approach has outperformed all of the handcraft feature based methods. In particular, the best performing architecture (Resnet-18) had significantly overcame the handcrafted detectors in both precision and recall metrics as shown in Figure 4. However, these improvements came at the cost of high detection time, with our proposed approach needing an extra 370ms to perform the detection. We notice also that all detection methods are too slow for real-time detection requirements.

Table 2 shows that the best compromise between detection performance and processing times has been obtained using the Faster R-CNN model. These findings are also confirmed by the results found in literature as shown by the works of Huang et al. [20] and Ren et al. [15]. However, we note that there are still many areas of possible improvements, such as the ratio of false positives. In fact, Figure 5 presents a sample of these false positives and compares their frequency in the results of both of our proposed detectors. We notice that the false positives are more common in the results of the handcrafted features based detector. These false positives are caused by the complexity of our real-world environment, which features a large amount of vertical elements such as trees, traffic signs, and light poles for example.



(a)

(b)

Figure 4. Comparison of detection results: (a) Proposed Handcraft features detector (b) Proposed Resnet-18 detector

(a)

(b)

Figure 5. Comparison of false positive presence: (a) Proposed Handcraft features detector (b) Proposed Resnet-18 detector

Figure 6. Example of missed detections caused by occlusions

Figure 6 highlights another limitation of our detectors, which is occlusion handling. Indeed, the database contains many scenarios where pedestrians move in groups, which creates a large number of occluded persons. This category of pedestrians is very difficult to detect, which causes a drop in our detectors performances. Another factor that explains the missed detections is the distance of certain pedestrians from the vehicle's camera. Beyond a distance of approximately 50 meters, these pedestrians appear in the camera view with small sizes below 40 pixels in height. These small sized pedestrians are challenging to detect.

5. CONCLUSION

In this paper, we propose to compare the performances of deep learning detectors and handcraft feature based detectors. To perform this comparison, we propose two different pedestrian detection frameworks: Our first detector uses a Faster R-CNN architecture trained using transfer learning techniques, while our second detector uses block matching and HOG features to detect pedestrians. The results show that our proposed deep detector is able to outperform handcrafted feature based detectors in detection performance while still

retaining superior robustness against false positives. However, handcraft feature based detectors are faster than their deep learning counterpart, which may be of interest in certain applications. The results produced by our detectors can be exploited in many applications in safety-critical applications such as intelligent transportation systems. In future work, further expansions to this work will be explored to further improve the results. One possible direction is to develop collaborative intelligence for roadside pedestrian detection.

REFERENCES

[1] Guerrero-Ibànez, J., Zeadally, S., Contreras-Castillo, J. (2018). Sensor technologies for intelligent transportation systems. Sensors, 18(4): 1212. http://dx.doi.org/10.3390/s18041212

[2] Mimouna, A., Alouani, I., Ben Khalifa, A., El Hillali, Y., Taleb-Ahmed, A., Menhaj, A., Ouahabi, A., Ben Amara, N.E. (2020). OLIMP: A heterogeneous multimodal dataset for advanced environment perception. Electronics, 9(4): 560. https://doi.org/10.3390/electronics9040560

[3] Jegham, I., Ben Khalifa A., Alouani I, Mahjoub M.A. (2019). MDAD: A multimodal and multiview in-vehicle driver action dataset. In: Vento M., Percannella G. (eds) Computer Analysis of Images and Patterns. CAIP 2019. Lecture Notes in Computer Science, 11679: 518-529. https://doi.org/10.1007/978-3-030-29888-3_42

[4] Jegham, I., Ben Khalifa, A., Alouani, I., Mahjoub, M.A. (2020). Vision-based human action recognition: An overview and real world challenges. Forensic Science International: Digital Investigation, 32: 200901. https://doi.org/10.1016/j.fsidi.2019.200901

[5] Brunetti, A., Buongiorno, D., Trotta, G.F., Bevilacqua, V. (2018). Computer vision and deep learning techniques for pedestrian detection and tracking: A survey. Neurocomputing, 300: 17-33. https://doi.org/10.1016/j.neucom.2018.01.092

[6] Kumar, K., Mishra, R.K. (2019). A robust mRMR based pedestrian detection approach using shape descriptor. Traitement du Signal, 36(1): 79-85. https://doi.org/10.18280/ts.360110

[7] Chebli, K., Khalifa, A.B. (2018). Pedestrian detection based on background compensation with block-matching algorithm. 2018 15th International Multi-Conference on Systems, Signals Devices (SSD), pp. 497-501. https://doi.org/10.1109/SSD.2018.8570499

[8] Yazdi, M., Bouwmans, T. (2018). New trends on moving object detection in video images captured by a moving camera: A survey. Computer Science Review, 28: 157-177. https://doi.org/10.1016/j.cosrev.2018.03.001

[9] Khalifa, A.B., Alouani, I., Mahjoub, M.A., Amara, N.E.B. (2020). Pedestrian detection using a moving camera: A novel framework for foreground detection. Cognitive Systems Research, 60: 77-96. https://doi.org/10.1016/j.cogsys.2019.12.003

[10] Cho, J., Jung, Y., Kim, D.S., Lee, S., Jung, Y. (2019). Moving object detection based on optical flow estimation and a gaussian mixture model for advanced driver assistance systems. Sensors, 19(14): 3217. http://dx.doi.org/10.3390/s19143217

[11] Zhang, J., Ding, Y., Xu, H., Yuan, Y. (2019). An optical flow based moving objects detection algorithm for the UAV. 2019 IEEE 4th International Conference on Computer and Communication Systems (ICCCS), pp. 233-238. https://doi.org/10.1109/CCOMS.2019.8821661

[12] Byeon, Y., Kwak, K. (2017). A performance comparison of pedestrian detection using faster RCNN and ACF. 2017 6th IIAI International Congress on Advanced Applied Informatics (IIAI-AAI), Hamamatsu, pp. 858-863. https://doi.org/10.1109/IIAI-AAI.2017.196

[13] Kong, W., Li, N., Li, T., Li, G., (2018). Deep pedestrian detection using contextual information and multi-level features, In International Conference on Multimedia Modeling, 10704: 166-177. https://doi.org/10.1007/978-3-319-73603-7_14

[14] Mateus, A., Ribeiro, D., Miraldo, P., Nascimento, J.C. (2019). Efficient and robust pedestrian detection using deep learning for human-aware navigation. Robotics and Autonomous Systems, 113: 23-37. https://doi.org/10.1016/j.robot.2018.12.007

[15] Ren, S., He, K., Girshick, R., Sun, J. (2017). Faster R-CNN: Towards real-time object detection with region proposal networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(6): 1137-1149. https://doi.org/10.1109/TPAMI.2016.2577031

[16] Yun, K., Lim, J., Choi, J.Y. (2017). Scene conditional background update for moving object detection in a moving camera. Pattern Recognition Letters, 88: 57-63. https://doi.org/10.1016/j.patrec.2017.01.017

[17] Yu, Y., Kurnianggoro, L., Jo, K.H. (2019). Moving object detection for a moving camera based on global motion compensation and adaptive background model. International Journal of Control, Automation and Systems, 17(7): 1866-1874. https://doi.org/10.1007/s12555-018-0234-3

[18] Heo, B., Yun, K., Choi, J.Y. (2017). Appearance and motion based deep learning architecture for moving object detection in moving camera. 2017 IEEE International Conference on Image Processing (ICIP), pp. 1827-1831. https://doi.org/10.1109/ICIP.2017.8296597

[19] Rozantsev, A., Lepetit, V., Fua, P. (2017). Detecting flying objects using a single moving camera. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(5): 879-892. https://doi.org/10.1109/TPAMI.2016.2564408

[20] Huang, J., Sun, C., Zhu, M., Korattikara, A., Fathi, A., Fischer, I., Wojna, Z., Song, Y., Guadarrama, S., Murphy, K. (2016). Speed/accuracy trade-offs for modern convolutional object detectors. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7310-7311. https://doi.org/10.1109/CVPR.2017.351

[21] Haralick, R., Shapiro, L. (1992). Computer and robot vision, ser. Computer and Robot Vision. Addison-Wesley Pub. Co.

[22] Dalal, N., Triggs, B (2005). Histograms of oriented gradients for human detection. 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), 1: 886-893. https://doi.org/10.1109/CVPR.2005.177

[23] Jegham, I., Ben Khalifa, A. (2017). Pedestrian detection in poor weather conditions using moving camera. 2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA), pp. 358-362. https://doi.org/10.1109/AICCSA.2017.35

[24] Khalifa, A.B. I2V-MVPD database, 2019, https://sites.google.com/site/benkhalifaanouar1/6-datasets, accessed on 12 December 2019.
[25] Dollar, P., Appel, R., Belongie, S., Perona, P. (2014). Fast feature pyramids for object detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 36(8): 1532-1545. https://doi.org/10.1109/TPAMI.2014.2300479