Fast car crash detection in video

V. Machaca Arceda*†, E. Laura Riveros†
*Universidad La Salle, †Universidad Nacional de San Agustín
{vicente.machaca.a, elian.laura.riv}@gmail.com

***Abstract*—In this work, we aim to detect car crash accidents in video. We propose a three-stage framework: The first one is a car detection method using convolutional neural networks, in this case, we used the net You Only Look Once (YOLO); the second stage is a tracker in order to focus each car; then the final stage for each car we use the Violent Flow (ViF) descriptor with a Support Vector Machine (SVM) in order to detect the car crashes. Our proposal is almost in real time with just 0.5 seconds of delay and also we got a 89% accuracy detecting car crashes.**

***Keywords*—ViF, SVM, Deep Learning, Convolutional Networks, Car Crash.**

I. Introduction

According to the World Health Organization (WHO) every year, 1.25 million people die as a result of road traffic crash, also between 20 and 50 million more people got injured. Moreover, 90% of road traffic deaths are in low-and middle-income countries. [13]. For example in Figure 1, we show the number of road traffic deaths by country, Indian and China have the highest mortality [12]. According to [16], the main reason for deads is the delay in reporting the accidents to hospitals and the delay in an ambulance reaching the car crash location.

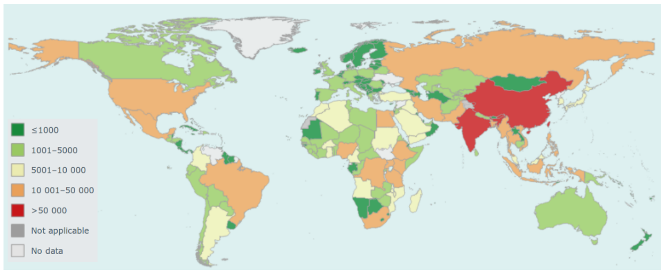


Figure 1: Number of road traffic deaths.

Technology could help alerting the drivers and helping the development of better-driven behaviors [9], [24]. Moreover, it is necessary a car crash reporting system in real time, there are sensors in vehicles and surveillance video systems in order to detect road accidents and send alerts, but still, there is a lot of research to do in order to have an accurate system. That's the reason we propose a framework based on deep learning and ViF descriptor, the main goal is to have an almost real-time system that could accurately detect car crashes.

This work is organized as follow: The first have presented an introduction, the next part will show the related works, the next part will present our proposal, the next part will present our result and finally we will present the conclusions.

II. Related Work

A lot of technologies have been applied in order to detect or avoid car accidents, such as sensors, Global System for Mobile (GSM), accelerometers, Global Positioning System (GPS), survilleance cameras, smartphones, etc. A recent work presented in [21] uses GSM and GPS with another hardware sensors that recognized crushes, when an accident occurs, the coordinates of the location of accident obtained by GPS, are sent via GSM network to user-defined mobile number. In the work of [23] is added an alcohol sensor and temperature sensor in order to detect fire. Also, in the work of [26] the accelerometer, GPS and GSM are used, but in this case, it is not installed in the car, they proposed the use of smartphones as a car crash detector. They also used the acoustic data to improve the accuracy.

Frameworks installed in smartphones with the role of driver assistant have also been research topic. In the work of [5], the authors proposed a driver assistant using smartphones, the app show traffic information and the occurrence of road incidents. Similarly, in the work of [7], is presented an app that collects information as: movement, maneuvering and steering wheel movement; then using a machine learning algorithm they predict the aggressive drivers. Moreover, in the work of [27] the smartphone's frontal camera is used in order to detect situations when the driver is tired or distracted.

In the other hand, there are systems that process the video of surveillance cameras, for example in [16], the authors proposed a three stages system in order to detect crushed cars, the first stage detect cars, then a texture descriptor with SVM is used to recognized crushed parts, finally, other classifier is used to detect car's parts in order to remove false positives, they got 81.83% of accuracy. In [10], the video frames are used in order to extract features such as the variation rate of the velocity, position, area, and direction of moving vehicles, then if the sum of variation of these features is greater than a threshold, the action is label as a car crush, regrettably they don't have enough metrics to measure their proposal. Also, in the work of [15], a Linear Discriminant Analysis (LDA) with SVM is used in order to recognized cars, also foreground object segmentation is carried out by Markov Random field (MRF) with Bayesian estimation process, the author track all the cars for accident analysis.

Also, in [25], a tracking algorithm is presented as weighted combination of low-level features extracted from moving vehicles and low-level vision analysis on vehicle regions extracted, the author achieved a 90-93% of detection rate and 88-92% of tracking rate, they used the velocity of each car in order to detect the accidents, for car crash detection, they got a precision of 87.5%. Moreover, in [22], analyzes traffic data and also, they propose a method for implement a real-time emergency vehicle detection system, they used Canny Edge Detector which tracks the red beacon signal.

Other methods as Bag of Visual Words (BoVW) is used for traffic scenes classification, as is in the case of [28], each video is encoded as a bag of SIFT feature vectors, the distribution is described by a Gaussian Mixture Models (GMM), the mean average precision on the TRECVID 2005 [14] dataset reported 60.4%

III. Proposal

A three-stage framework is proposed, the first stage finds cars involved in the scene, here we have used YOLO. In the second stage, we track each one saving their position for a period of time. The results are some short videos for each car. Finally, in the last stage, we take the short videos from the second stage and compute ViF algorithm, previously trained with a SVM classifier, in order to detect car crashes. Figure 2 illustrates the proposed three-stage model.

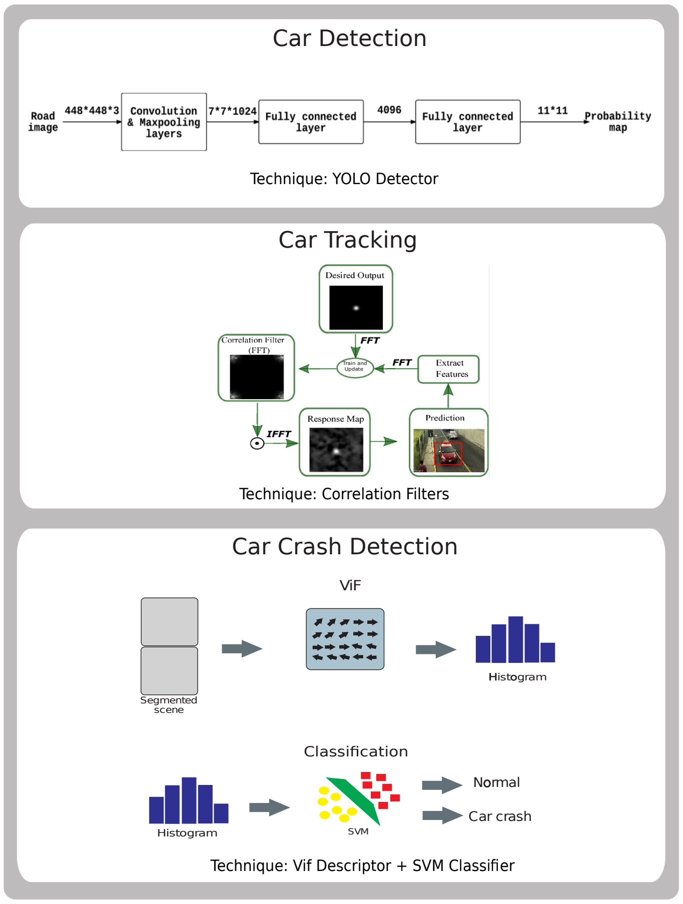
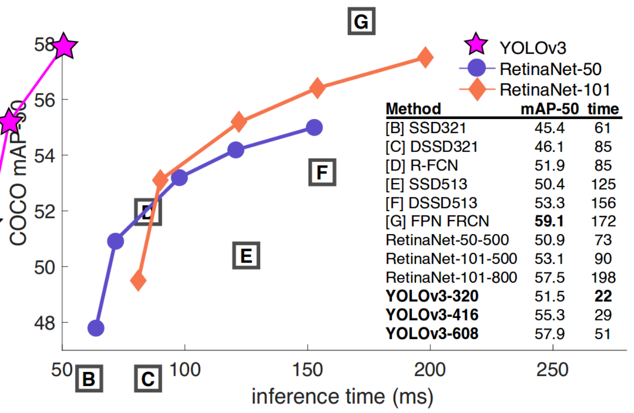


Figure 2: The proposed model with a simplified structure of each stage, [29], [3].

A. Car detection

For car detection we used the YOLO net, it was proposed by [18], improved in [19] and more recently, the last version was presented in [20]. Despite other convolutional networks, it detects objects with a single pass creating a grid of SxS boxes, where each box has a logistic regression and a classification method, the regression method predict each box with five values: x, y, w, h and the confidence of the object being there. The classifier predicts C conditional class probabilities. At the final stage multiple bounding boxes appear around a single object, so non-maximum suppression is applied in order to keep the strongest detection around a single object. The architecture of YOLO for this research project is showed in the table I, obtained from Darknet library [17]. The result in this step are the bounding boxes for each car, as shown in Figure 4.

In Figure 3, the YOLO performance is showed, here, this network has a very high accuracy and more important have a very low time processing against the rest.



Method	mAP-50	time
[B] SSD321	45.4	61
[C] DSSD321	46.1	85
[D] R-FCN	51.9	85
[E] SSD513	50.4	125
[F] DSSD513	53.3	156
[G] FPN FRCN	**59.1**	172
RetinaNet-50-500	50.9	73
RetinaNet-101-500	53.1	90
RetinaNet-101-800	57.5	198
YOLOv3-320	51.5	**22**
YOLOv3-416	55.3	29
YOLOv3-608	57.9	51

Figure 3: Comparison of YOLO and other networks, speed/accuracy trade-off on the mAP at .5 IOU metrics. Taken from: [20]

B. Car tracking

We apply a car tracking algorithm each 30 frames, generating one tracker per car and saving the tracker position. Then we extract small videos, one per car. In the right of the Figure 5 we show an example, where three small videos have been extracted because three cars were detected in the first stage of the proposed model.

The car tracker used in our model is based in a visual

Table I: Architecture of YOLO

Inicialización de pesos	kernel	stride	output shape
Input			(416, 416, 3)
Convolution	3x3	1	(416, 416, 16)
MaxPooling	2x2	2	(208, 208, 16)
Convolution	3x3	1	(208, 208, 32)
MaxPooling	2x2	2	(104, 104, 32)
Convolution	3x3	1	(104, 104, 64)
MaxPooling	2x2	2	(52, 52, 64)
Convolution	3x3	1	(52, 52, 128)
MaxPooling	2x2	2	(26, 26, 128)
Convolution	3x3	1	(26, 26, 256)
MaxPooling	2x2	2	(13, 13, 256)
Convolution	3x3	1	(13, 13, 512)
MaxPooling	2x2	1	(13, 13, 512)
Convolution	3x3	1	(13, 13, 1024)
Convolution	3x3	1	(13, 13, 1024)
Convolution	1x1	1	(13, 13, 125)

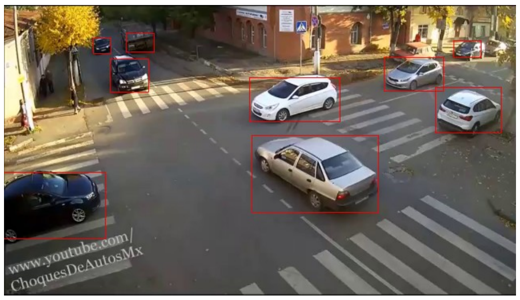


Figure 4: Car detection with YOLO

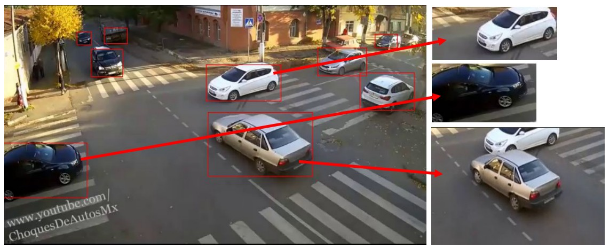


Figure 5: Video extraction for each tracker (car). The left side shows a surveillance video, and the right side shows three small videos, each one generated per car. Each small video has different sizes but all of them have 30 frames.

objects algorithm with correlation filters, proposed by Danelljan et al. [4]. The algorithm consists in the translation and scale estimation of the object through a series of correlations over the Fourier domain, where each correlation filter is submitted to an online learning process..

The procedure of correlation filters since the position of the visual object, according to Chen et al. [3], can be summarized as follows.

In each frame the region whose position was predicted in the previous frame is extracted for detection. Then the features HOG [4] are extracted since the intensity level of each pixel of the image region, a cosine window [2] is applied for smoothing a boundary effects. Next, a serie of correlation operations are performed with element-wise multiplications using the Discrete Fourier Transform (DFT), it is computated with an efficient Fast Fourier Transform (FFT) algorithm, the answer is a spatial confidence map obtained with the inverse of FFT (IFFT). The position with the maximum value located on the map is predicted as the new position of the target. Then, the features of the estimated position region are extracted again for the training and updating of the correlation filter, being part of the online learning, the IFFT is applied again to obtain the response map and another prediction, this process continous for each frame, Fig. 6

The basic mathematical description of the procedure is given from the Convolution Theorem where the element-wise multiplication is perform in the frequency domain, it can be seen in the equation 1, where $\otimes$ is the expected correlation between the input region x and the filter h , the IFFT operation is denoted as $textitF^{-1}$ from the element-wise multiplication $\odot$ from $\hat{x}^{'}$ and $\hat{h}^{*}$ computated with the Fourier Transform.

$$x \otimes h = \mathscr{F}^{-1}(\hat{x} \odot \hat{h}^{*}) \tag{1}$$

The expected output of $x \otimes h$ is represented with the variable y, and the new instance x is $x^{'}$, as the equation 2.

$$y = \mathscr{F}^{-1}(\hat{x}^{'} \odot \hat{h}^{*}) \tag{2}$$

finally the correlation filter is represented by a wise-element division, equation 3, from this step the training and updating is carried out.

$$\hat{h}^{*} = \frac{\hat{y}}{\hat{x}^{'}} \tag{3}$$

C. Car crash detection

In order to detect a car crash scene, we are going to use the ViF descriptor because of the very low cost and acceptable accuracy. The ViF descriptor regards the statistics of magnitude changes of flow vectors over time as we see in Figure 7. In order to get these vectors, [6] used the optical flow algorithm proposed by [11] named Iterative Reweighted Least Squares (IRLS), but in this context, we used the ViF descriptor with Horn-Schunck [8] as optical flow algorithm proposed by [1].

The ViF descriptor is presented in algorithm 1, here we get a binary, magnitude-change, significance map b_t for each frame f_t. Then we get a mean magnitude-change map, for each pixel, over all the frames with the equation 4:

$$b_{x,y} = (1/T) \sum_{t} b_{x,y,t} \tag{4}$$

Then the ViF descriptor is a vector of frequencies of quantized values $b_{x,y}$. For more details you could see the work of [6].

Then the car crash detector is trained using a SVM classifier with a polynomial kernel, taking as input the result of ViF

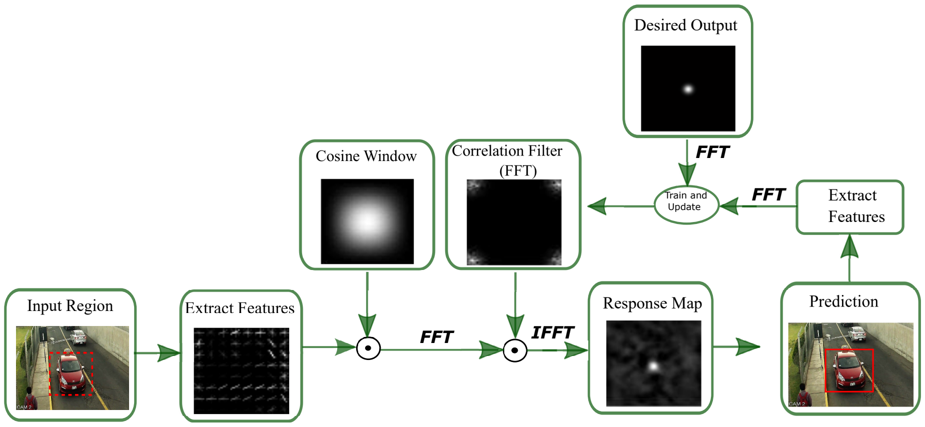


Figure 6: General workflow for a correlation filter method, based on [3].

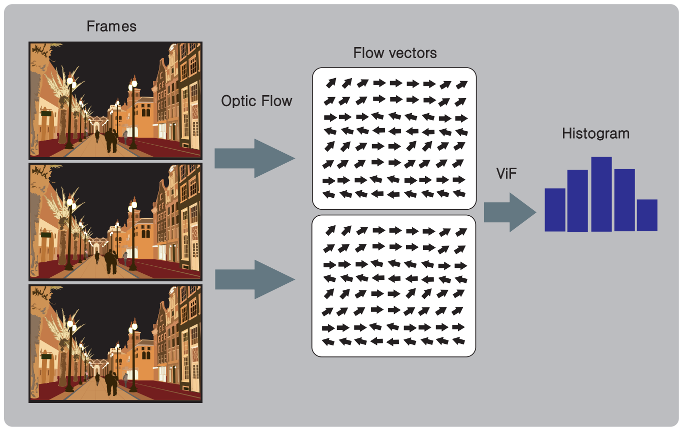


Figure 7: ViF descriptor in video.

Algorithm 1 ViF descriptor

Data: S = Sequence of gray scale images. Each image in S is denoted as $f_{x,y,t}$, where $x = 1, 2, ..., N$, $y = 1, 2, ..., M$ and $t = 1, 2, ..., T$.
Result: Histogram($b_{x,y}$; n_bins = 336)

```
1: for  $t = 1$ to $T$ do
2:     Get optical flow $(u_{x,y,t}, v_{x,y,t})$ of each pixel $p_{x,y,t}$
       where $t$ is the frame index.
3:     Get magnitude vector: $m_{x,y,t} = \sqrt{u^2_{x,y,t} + v^2_{x,y,t}}$
4:     For each pixel we get:
       $b_{x,y,t} = \begin{cases} 1 & \text{if } |m_{x,y,t} - m_{x,y,t-1}| >= \theta \\ 0 & \text{other case} \end{cases}$
       where $\theta$ is a threshold adaptively set in each frame to the
       average value of $|m_{x,y,t} - m_{x,y,t-1}|$.
```

descriptor. In the experiments we use cross-validation with k=10. The whole method to detect car crash in video is shown in Figure 8

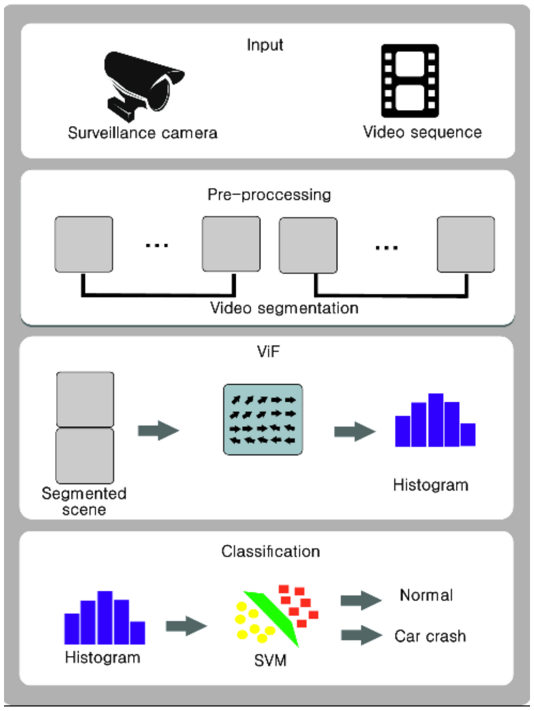


Figure 8: The car crash detection method.

IV. Experiment and Results

A. Datasets

Actually, there is no a good car-crash video dataset, for that reason we have built our own dataset, Car Crash Videos (CCV). We are regarding videos from surveillance cameras for the experiments because the device (camera) has a static position, and surveillance videos are easy to collect from different web pages of videos.

As we mentioned, we used the ViF descriptor with a SVM classifier on each car after car detector YOLO. For that reason we could no train our classifier with each entire video in dataset, instead we generate small videos, as you can see in Figure 5. As the result of new small videos generation, we got 120 normal video traffic and 57 car-crash videos, each video is 30 frames. In order to have a balanced dataset we just considered 57 normal video traffics, so in total we got 114 videos, these videos compound the CCV dataset.

In Figure 9, we could see some video frames of car crashes, in Figure 10 we see some video frames of normal traffic.

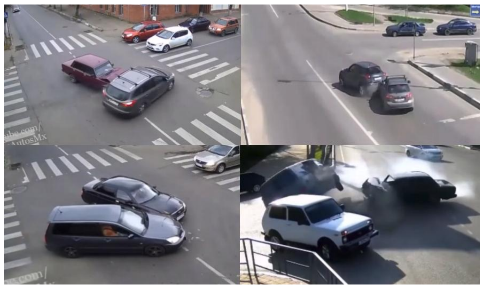


Figure 9: Some frames of CCV dataset for car crashes.

Figure 10: Some frames of CCV dataset for normal traffic.

B. Results

In Table II we show the accuracy, AUC, recall and precision. We can see that the recall is 0.8. That means, we have a 80% probability of detect just car crashes. In the Figures 11 and 12 the ROC curve and the precision-recall curve are presented. Despite the small dataset CCV, we have good results. Moreover, it is necessary to highlight that the dataset contains surveillance videos in the worst conditions, bad illumination, shadows, poor resolution, and the presence of wind. These conditions are a challenge for any video on image processing.

Table II: The performance of ViF on each car. The Accuracy (ACC) of the classifier were evaluated by cross-validation (k=10) and also the Area Under the Curve (AUC) is included with the Precision, Recall and the Average precision-recall.

Metric	Value
Accuracy	0.75
AUC	0.76
Recall	0.80
Precision	0.66
Average precision-recall	0.81

We also measure the time processing, in this case avoiding the time for car detection we just get a 2.15 seconds for processing a video of 2 seconds. This is the main reason we use ViF because of its very low cost. It is shown in Table III. The measurement was evaluated in a computer with a 1.8 GHz processor.

Table III: Time processing of ViF for car crash detection.

	Video duration (seg.)	Time processing (seg.)
ViF	2	2.1563

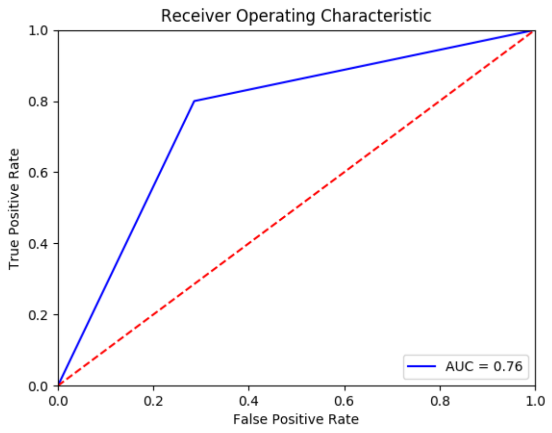


Figure 11: ROC curve of ViF performance.

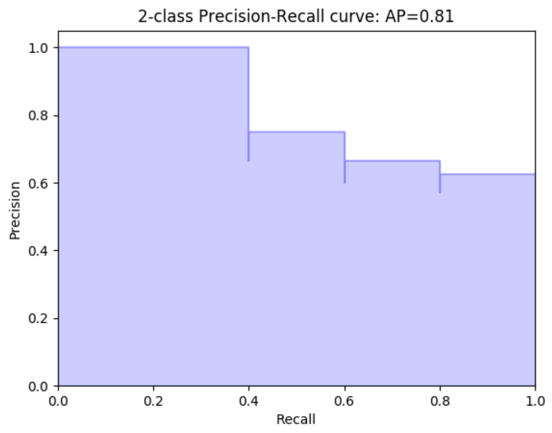


Figure 12: Precision-Recall curve.

In Figure 13, we show our result in surveillance videos, as you can see, we mark the exact position of car crashes. We know that exact position thanks the tracker position in the scene. Also, in the top right corner of the Figure 13 there are two red boxes because we paint a red box for each car in the collision, sometimes just one car is detected, but it is enough in order to detect car crashes.

V. Future work

We got good results, but it is not enough, according to ViF descriptor we just consider the optic vector magnitude,

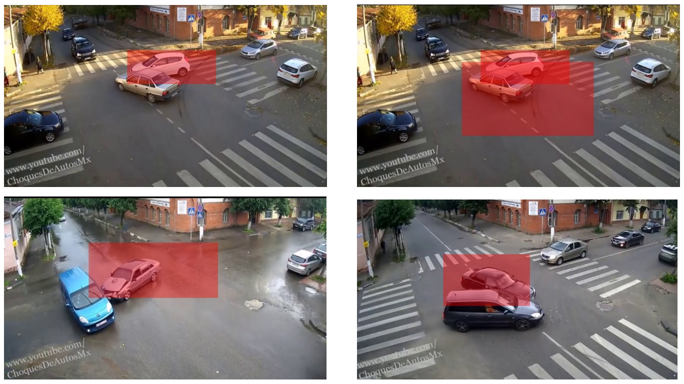


Figure 13: Car crash detection with the proposed method.

but in car crash detection, also, the vector direction could be relevant. For that reason in future work we are planning to use these features in order to improve our accuracy.

Moreover, we have built a CCV dataset with more than 100 videos. We are planning to improve this dataset, regarding surveillance videos for the experiments.

VI. Conclusions

We proposed a model based on car detection using YOLO, then, we track each car with correlation filters algorithm and finally we used the ViF descriptor on each tracker in order to detect a car crash.
ViF have good results and moreover, a very low processing time, that's the main reason we used it for a very fast car crash detection.
In this case we have used only the optic flow magnitude but we believe that the direction have relevant information, in future works we are going to used these features in order to improve the performance.

References

[1] V. M. Arceda, K. F. Fabián, and J. Gutiérrez, "Real time violence detection in video," *IET Conference Proceedings*, pp. 6 (7 .)–6 (7 .)(1), Apr 2016. [Online]. Available: http://digital-library.theiet.org/content/conferences/10.1049/ic.2016.0030

[2] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 2544–2550.

[3] Z. Chen, Z. Hong, and D. Tao, "An experimental survey on correlation filter-based tracking," *arXiv preprint arXiv:1509.05520*, 2015.

[4] M. Danelljan, G. Häger, F. Khan, and M. Felsberg, "Accurate scale estimation for robust visual tracking," in *British Machine Vision Conference, Nottingham, September 1-5, 2014*. BMVA Press, 2014.

[5] S. Diewald, A. Möller, L. Roalter, and M. Kranz, "Driveassist: a v2x-based driver assistance system for android," in *Konferenz Mensch & Computer: 09/09/2012-12/09/2012*. Oldenburg Wissenschaftsverlag, 2012, pp. 373–380.

[6] T. Hassner, Y. Itcher, and O. Kliper-Gross, "Violent flows: Real-time detection of violent crowd behavior," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*, June 2012, pp. 1–6.

[7] J.-H. Hong, B. Margines, and A. K. Dey, "A smartphone-based sensing platform to model aggressive driving behaviors," in *Proceedings of the 32nd annual ACM conference on Human factors in computing systems*. ACM, 2014, pp. 4047–4056.

[8] B. K. Horn and B. G. Schunck, "Determining optical flow," in *1981 Technical symposium east*. International Society for Optics and Photonics, 1981, pp. 319–331.

[9] J. Jiménez-Pinto and M. Torres-Torriti, "Optical flow and driver's kinematics analysis for state of alert sensing," *Sensors*, vol. 13, no. 4, pp. 4225–4257, 2013.

[10] Y.-K. Ki, "Accident detection system using image processing and mdr," *International Journal of Computer Science and Network Security IJCSNS*, vol. 7, no. 3, pp. 35–39, 2007.

[11] C. Liu, "Beyond pixels: exploring new representations and applications for motion analysis," Ph.D. dissertation, Citeseer, 2009.

[12] W. H. Organization, "Global health observatory (gho) data," http://www.who.int/gho/road_safety/mortality/traffic_deaths_number/en/, 2018, [Online; accessed 04-27-2018].

[13] ——, "Road traffic injuries," http://www.who.int/en/news-room/fact-sheets/detail/road-traffic-injuries, 2018, [Online; accessed 04-27-2018].

[14] P. Over, T. Ianeva, W. Kraaij, and A. F. Smeaton, "Trecvid 2005-an overview," 2005.

[15] M. Rajalakshmi, "Road side video surveillance in traffic scenes using map-reduce framework for accident analysis," *Biomedical Research*, 2016.

[16] V. Ravindran, L. Viswanathan, and S. Rangaswamy, "A novel approach to automatic road-accident detection using machine vision techniques," *INTERNATIONAL JOURNAL OF ADVANCED COMPUTER SCIENCE AND APPLICATIONS*, vol. 7, no. 11, pp. 235–242, 2016.

[17] J. Redmon, "Darknet: Open source neural networks in c," http://pjreddie.com/darknet/, 2013–2016.

[18] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[19] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," *arXiv preprint arXiv:1612.08242*, 2016.

[20] ——, "Yolov3: An incremental improvement," *arXiv*, 2018.

[21] N. H. Sane, D. S. Patil, S. D. Thakare, and A. V. Rokade, "Real time vehicle accident detection and tracking using gps and gsm," *International Journal on Recent and Innovation Trends in Computing and Communication ISSN*, pp. 2321–8169.

[22] N. Sankhe, P. Sonar, and D. Patel, "An overview of image processing for traffic applications," *IOSR Journal of Engineering*, vol. 4, no. 4, pp. 08–14, 2014.

[23] D. Shah, R. Nair, V. Parikh, and V. Shah, "Accident alarm system using gsm, gps and accelerometer," *International Journal of Innovative Research in Computer and Communication Engineering*, vol. 3, no. 4, 2015.

[24] R. Tapia-Espinoza and M. Torres-Torriti, "Robust lane sensing and departure warning under shadows and occlusions," *Sensors*, vol. 13, no. 3, pp. 3270–3298, 2013.

[25] L. VASU, "An effective step to real-time implementation of accident detection system using image processing," Master's thesis, Oklahoma State University, 12 2010.

[26] J. White, C. Thompson, H. Turner, B. Dougherty, and D. C. Schmidt, "Wreckwatch: Automatic traffic accident detection and notification with smartphones," *Mobile Networks and Applications*, vol. 16, no. 3, p. 285, 2011.

[27] C.-W. You, M. Montes-de Oca, T. J. Bao, N. D. Lane, H. Lu, G. Cardone, L. Torresani, and A. T. Campbell, "Carsafe: a driver safety app that detects dangerous driving behavior using dual-cameras on smartphones," in *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*. ACM, 2012, pp. 671–672.

[28] X. Zhou, X. Zhuang, S. Yan, S.-F. Chang, M. Hasegawa-Johnson, and T. S. Huang, "Sift-bag kernel for video event analysis," in *Proceedings of the 16th ACM international conference on Multimedia*. ACM, 2008, pp. 229–238.

[29] Y. Zhou, H. Nejati, T.-T. Do, N.-M. Cheung, and L. Cheah, "Image-based vehicle analysis using deep neural network: A systematic study," in *Digital Signal Processing (DSP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 276–280.