IEEE Access
Multidisciplinary ⁝ Rapid Review ⁝ Open Access Journal

Received June 5, 2021, accepted June 20, 2021, date of publication June 28, 2021, date of current version July 12, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3092938

Vehicle License Plate Detector in Compressed Domain

MUHAMMET SEBUL BERATOĞLU AND BEHÇET UĞUR TÖREYİN
Signal Processing for Computational Intelligence Research Group, Informatics Institute, İstanbul Technical University, 34469 İstanbul, Turkey

Corresponding author: Muhammet Sebul Beratoğlu (beratoglu@itu.edu.tr)

This work was supported in part by the Scientific Research Projects Coordination Department (BAP), Istanbul Technical University, under Project ITU-BAP MGA-2017-40964, and in part by The Scientific and Technological Research Council of Turkey (TÜBİTAK) - TEYDEB under Grant 3190538.

ABSTRACT Data compression techniques allow data size to be reduced prior to data transmission and involve decompression upon transfer. This study shows for the first time that license plate (LP) detection can be accomplished without full decompression of the encoded data. Therefore, by determining in advance which images are required for LP recognition, computational costs of the system can be reduced. The proposed approach is realized on High Efficiency Video Coding (HEVC) based compressed video sequences. Two methods are provided that generate images from HEVC attributes. Fully decoded pixel domain images are also generated for comparative purposes from the same encoded data. The YOLO V3 Tiny Object Detector is used in order to detect LPs in the generated images. EnglishLP, a public dataset, is used to interpret the findings in terms of speed and precision and for comparison with previous studies. An additional contribution of the paper is that a new compressed domain LP database has been created and made publicly available, comprising images captured by a commercial license plate recognition system. Using at least two-orders-of-magnitude less amount of data, the proposed compressed domain LP detector achieved similar precision and recall values to those of the state-of-the-art LP detection schemes tested on both datasets. Moreover, the proposed method results in more than 30% saving in inference time. The results suggest that the proposed method can be utilized for rapid video archive searching applications.

INDEX TERMS Compressed domain image/video analysis, H.265, license plate detection, YOLO.

I. INTRODUCTION

With the rapid development of deep learning-based methods, object detection accuracy has increased in almost all areas of image processing, including license plate recognition (LPR) [2], [3]. Deep learning networks give high performance whereas they require high processing power. To be able to utilize computationally demanding deep learning-based methods, there are two different ways. The first way is to have high-capacity processors at the source of data and do processing in real-time. The second way, is to transfer data to a center and process it there with powerful computers. The first way increases the cost of the image processing unit, whereas, the second way increases the cost of data transmission and includes risks such as disconnection. For both ways, all efforts to reduce the amount of data to be processed or transmitted are especially important.

The aim of this research is to develop a faster vehicle license detection (LP) method using data available in a compressed domain (CD). Data compression techniques [4], [5] reduce the size of the data and make it suitable for network transmission. When data is received, the traditional method is to decompress it first, then perform any necessary analyses, such as LPR. Our strategy is to use partial decompressed data to detect LPs, as illustrated in Fig. 1. The underlying assumption is that encoded data preserves the discriminative properties of an LP. We provide methods for detecting LP using those properties. As a result, data can be processed faster or with less processing power. The possibility of lowering computing costs is very appealing, due to the wide variety of applications of LPR technology.

To the best of the authors' knowledge, no previous research has been conducted on LP detection in compressed domain data. We use High Efficiency Video Coding (HEVC) for video compression. We developed two different methods to represent LP characteristics inside encoded data. We use a

The associate editor coordinating the review of this manuscript and approving it for publication was Mauro Gaggero.

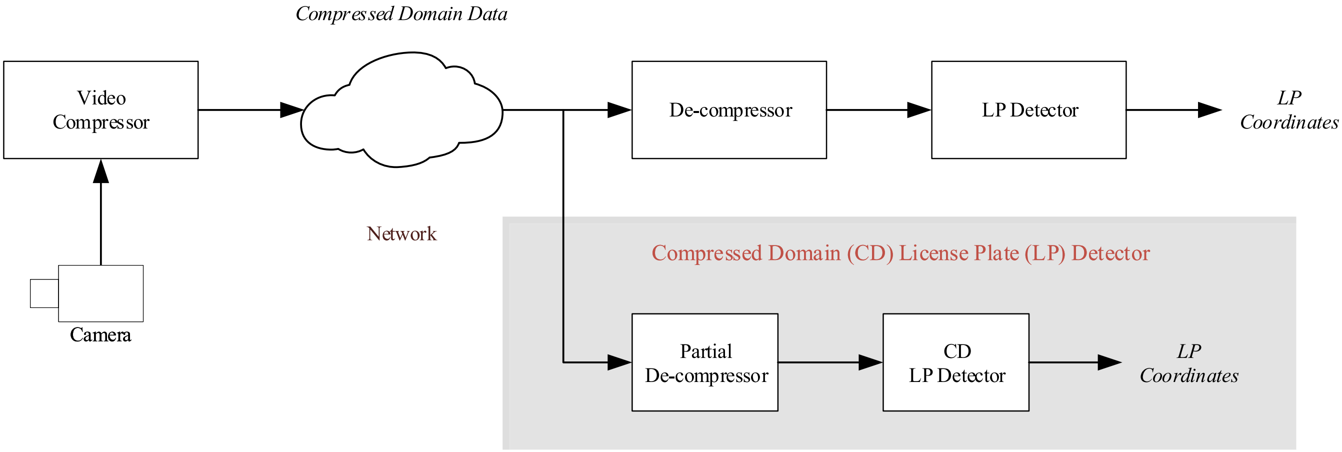

FIGURE 1. The general framework for LP detection in the pixel and compressed domains.

state-of-the-art object detector, YOLO, and train three different convolutional neural networks (CNNs) to show and compare our LP detection performance. The improved performance in terms of speed is demonstrated, while maintaining accuracy comparable to fully decoded data.

The paper is organized as follows. Section 2 discusses related work on LP detection and compressed domain analytics. HEVC and its attributes are briefly defined in Section 3. In Section 4, we introduce our compressed domain LP detection methodology. In Section 5, a new database for LP detection is introduced. Section 6 summarized the results obtained in terms of speed and accuracy. Section 7 is devoted to future works and discussions. Finally, in Section 8, the conclusions are drawn.

II. RELATED WORK

There has been many algorithms developed for LP detection [8]. Edge detection [9], [10], Gabor filters [12], SIFT features [11] and connected component analyses [13] are some of the common techniques that are used for traditional LP detection.

With the advance of deep learning, CNNs are successfully used for LP detection and high accuracy is achieved [2]. Delmar *et al.* proposed a CNN for detecting LPs, which calculates a score for each image sub-region to detect the region [14]. Hedry *et al.* used YOLO [31] as CNN and has achieved 98.22% accuracy on LP of Taiwan's cars [15]. Wanwei *et al.* train a light multi-task CNN (MTCNN) to detect LP for Chinese license plates [16]. Laroca *et al.* proposed an efficient and layout-independent Automatic License Plate Recognition (ALPR) system based on the cutting-edge YOLO object detector. They reported 99.92% vehicle detection recall accuracy, 99.51% LP detection, and a 96.8% average end-to-end recognition rate [17]. Min *et al.* utilize YOLOv2 for detection. They use k-means++ clustering algorithm to select the best number and size of plate candidate boxes and based on this information they modify the structure and depth of YOLOv2 model [18]. Tao *et al.* compared YOLO and SSD [4] in terms of LP detection and reported that YOLO achieved better accuracy [19]. Unlike these studies, we do detection in the compressed domain.

There have been many studies performing video analytics in the compressed domain [21]–[26]. Toreyin, uses Wavelet and Markov model for smoke detection in MJPEG2000 compressed video [27]. Bombardelli *et al.* tracked objects in H.264 encoded video [28]. Zhao *et al.* proposed an approach for real-time object tracking in H.265 [29]. Alvar *et al.* conducted a study to detect face localization in encoded data [30].

III. BACKGROUND

HEVC and its attributes used in our system are briefly defined in this section and the reason why we choose these attributes.

A. HIGH EFFICIENCY VIDEO CODING (HEVC)

High efficiency video coding (HEVC), also known as H.265, is a video compression format which is designed as a successor to the previous H.264 video compression format. H.264 is the most commonly used video coding standard worldwide [4]. HEVC, compared to H.264, can achieve from 25% to 50% better data compression at the same level of video quality [5]. HEVC is becoming one of the new video standards. Numerous IP camera manufacturers now include HEVC support by default.

B. HEVC ATTRIBUTES

Our compressed domain LP detection approach utilizes the following attributes of the HEVC. In doing so, compressed bit-stream is not required to be fully re-constructed. Therefore, data to be processed is less in amount.

1) BLOCK PARTITIONING STRUCTURE OF HEVC

HEVC encodes data using a flexible partitioning structure [5]. The image is divided into partitions, namely Coding Tree Units (CTU). The default size of CTU is 64 × 64 pixels.

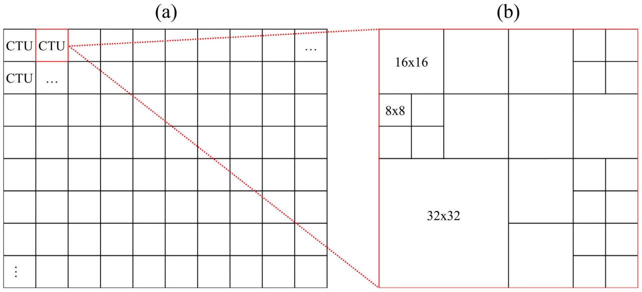

FIGURE 2. The HEVC block partitioning structure. a) The division of an image into CTUs of equal size. b) Subdivision of a CTU into CUs.

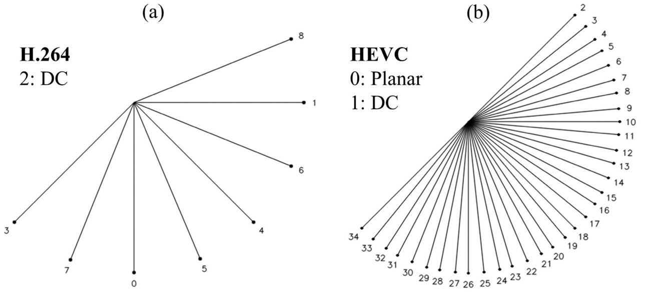

FIGURE 3. Intra prediction directions. a) The nine intra prediction modes of H.264. b) The thirty five intra prediction modes of H.265 (HEVC).

Fig. 2(a) shows how an image is segmented into CTUs of equal size. CTUs are divided into different sized Coding Units (CUs) depending on the complexity of the encoded region. CUs can be 8 × 8, 16 × 16, 32 × 32 or 64 × 64. CUs of various sizes are portrayed in Fig. 2(b).

We choose this attribute because the HEVC partitioning structure is designed to use small-sized CUs when encoding the complex texture of the image. In other words, high-band spatial content in the pixel domain requires using smaller CUs. There is an image of a vehicle in Fig. 4a and the corresponding block partitioning image is present in 4b. High-band spatial frequency areas, such as the LP zone, can be seen to be partitioned using smaller CU blocks. For LP detection in the compressed domain, this characteristic partitioning of LP regions is exploited.

2) PREDICTION UNITS OF HEVC

Each coding unit (CU) is divided into one or more Prediction Units (PUs) using Intra Prediction or Inter Prediction. Our method, uses only intra predicted frames to detect plate region. Once a plate region is detected, it can be tracked using inter prediction attributes until another intra-predicted frame arrives.

In an intra-predicted frame, each PU estimates from adjacent image data within the same image using DC prediction, planar prediction, and directional prediction. In the H.264 standard there are 8 directions defined to be used in prediction (cf. Fig. 3(a)). In the HEVC there are 33 different directions in addition to DC and planar predictions (cf. Fig. 3(b)).

The PUs, in intra-coded stream, are calculated from adjacent image data. Thus, PUs hold the correlation information between CUs. We choose PUs as a distinctive attribute, to create an image that reflects the correlation between pixels inside a plate region.

IV. COMPRESSED DOMAIN LICENSE PLATE (LP) DETECTOR

In this section, we describe our compressed domain LP detection methodology. First, we'll go over how the HEVC attributes are turned into images, namely HEVC images. Then, using HEVC images, we demonstrate how LP detection is performed.

A. IMAGE GENERATION FROM HEVC ATTRIBUTES

Our objective is to detect LPs without fully decoding the HEVC stream. The first step is to convert the encoded stream to an LP-detectable format. We select "image format" to represent HEVC attributes and generate images from the encoded stream. There are two advantages to using an image as an output. To begin, image processing techniques can be used to detect LPs. Second, the images produced aid in data interpretation. The two developed methods for constructing an image using HEVC attributes are described in the following sub-sections.

1) IMAGE GENERATION FROM BLOCK PARTITION STRUCTURE

The first method generates an image using HEVC block partition (BP) structure. Let $I(x, y)$ be the intensity value of an intra-coded image I at location (x, y). Let A be the set of boundary pixel locations of coding units (CUs) corresponding to the image I. An image I is converted into a binary image using (1).

$$f_{bp}(x, y) = \begin{cases} 1 & \text{if } (x, y) \in A \\ 0 & \text{else} \end{cases} \tag{1}$$

Pixels that cross CU boundaries are converted to white pixels, while the rest are converted to black pixels. The first method's output image will be referred to as HEVC Block Partition Image, abbreviated H_{bp}. An example of a H_{bp} is shown in Fig. 4.[1]

2) IMAGE GENERATION FROM PREDICTION UNIT INFORMATION

The second method generates an image using HEVC prediction unit (PU) information. An image I is converted into a gray-level image based on its PU values using (2).

$$f_{pu}(x, y) = \begin{cases} \alpha P(x, y) + \beta & \text{if } |C[I(x, y)]| = 8 \times 8 \\ \gamma & \text{else if } (x, y) \in A \\ 0 & \text{else} \end{cases} \tag{2}$$

[1] An earlier study of the H_{bp} based method was presented in Turkish [1].

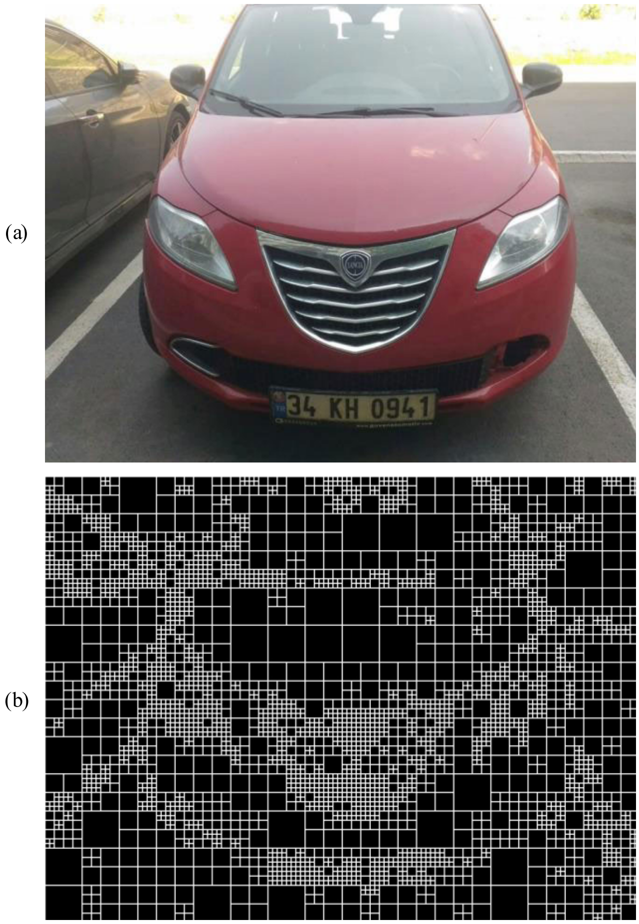

FIGURE 4. A sample H_{bp} image created using the block partition structure.

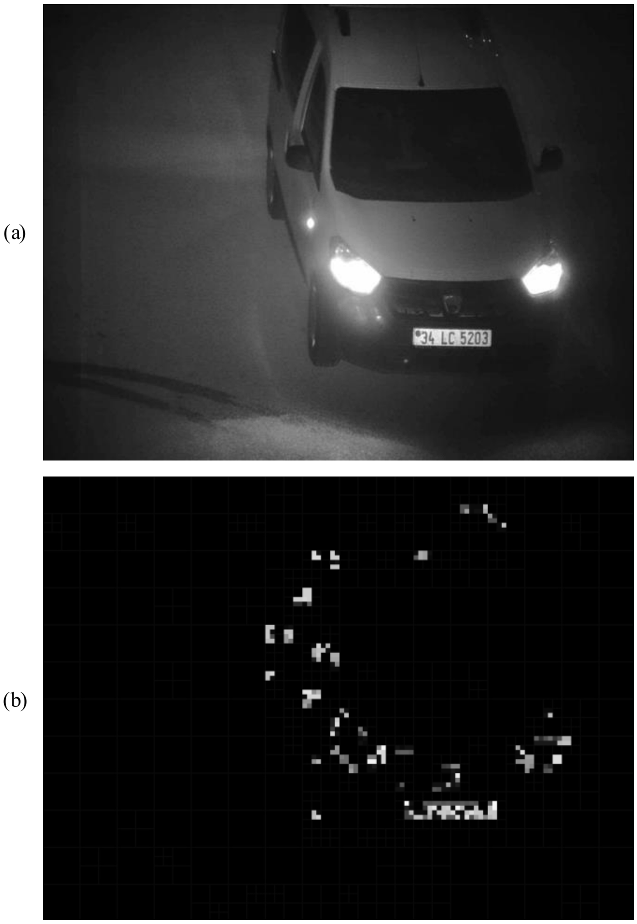

FIGURE 5. A sample H_{pu} image created using the prediction unit information.

where $P(x, y)$ denotes the prediction unit value for a pixel in an image I at location (x, y). And $C[I(x, y)]$ is the coding unit block corresponding to the image I at location (x, y). The $P(x, y)$ are integers in the range of 0 to 34 (cf. Fig. 3). A linear equation is used to convert PU values to the 0-255 pixel range, with α and β determined empirically. The β is used to distinguish PU values from the black background image. $\beta = 45$ is sufficient to generate a background difference, whereas $\alpha = 6$ generates distinct areas for various PU values. Apart from PU values, the generated image also preserves the block partition structure by using the γ parameter. For γ; the average value between the background and the minimum PU value is chosen, which is $\gamma = \frac{\beta}{2} = 24$. Typically, the LP will be located in small-sized CUs. Therefore, the output image is created in a way that only 8 × 8 sized CUs are visible. The remaining image area is set to black.

The image produced by the second method is referred to as HEVC Prediction Unit Image, abbreviated H_{pu}. A H_{pu} is illustrated in Fig. 5.

The final output of the HEVC decoding phase is a standard image, which we refer to as the pixel domain image, abbreviated P. The outputs of three image generation methods, namely Pixel Image (P), HEVC Block Partition Image (H_{bp}), and HEVC Prediction Unit Image (H_{pu}), are shown in Fig. 6.

When creating HEVC images, each CU is represented by a single pixel. Due to the fact that the smallest CU contains 8 × 8 pixels, the resulting image is 8 × 8 times smaller than the original. Working with HEVC images has the advantage of having condensed information, which results in faster detection. The size difference between the original image and the one constructed from HEVC attributes is shown in Fig. 7.

B. LP DETECTION

LP detection is one of the fundamental steps of license plate recognition. We use a CNN to detect LP, specifically the YOLO object detection method [31], [32]. YOLO is a popular object detection method optimized for real-time operation [31], [32]. YOLO has reported good precision and recall rates besides its fast execution times ((around 70 FPS) (76.8% mAP over the PASCAL-VOC dataset)). In many recent works, YOLO is used for real-time LP detection [15], [18]–[20]. In this work, a smaller version of YOLO, namely YOLOv3-tiny, is used to achieve even faster execution times while maintaining detection performance.

Fig. 8 illustrates our LP detection methodology in the HEVC domain. Our primary input is a HEVC bit-stream. This input is decoded into two distinct image types. The first is a

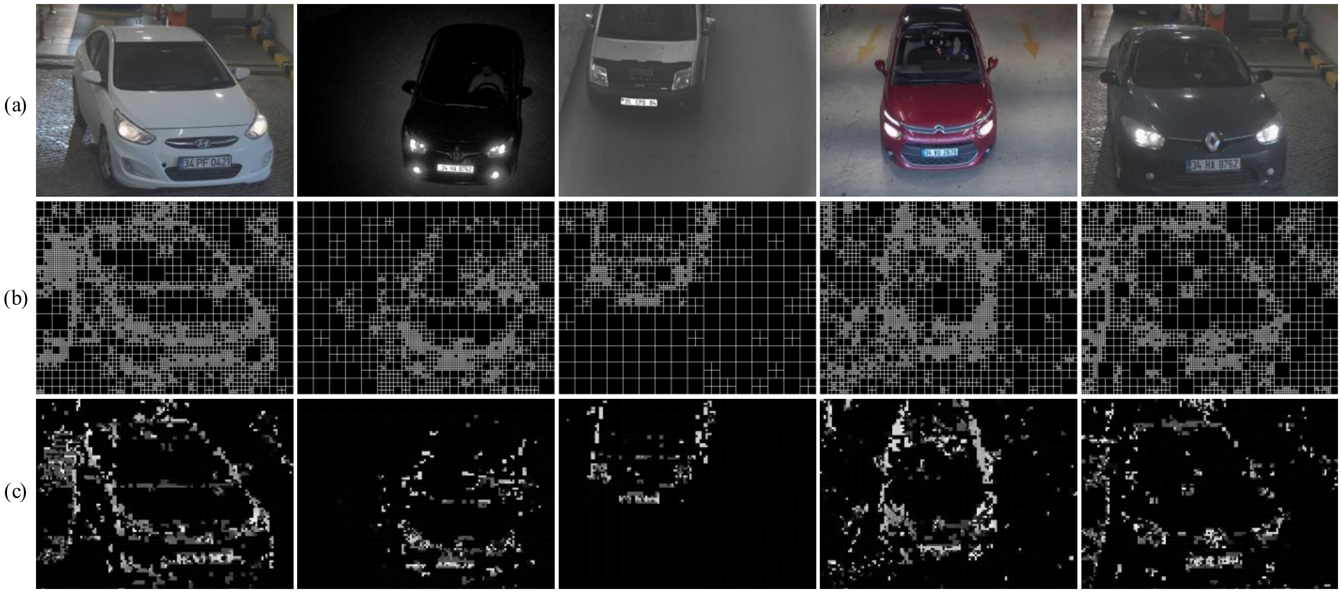

FIGURE 6. Three distinct image types produced from a variety of HEVC streams. a) Pixel domain (*P*) images. b) HEVC block partition (H_{bp}) images. c) HEVC prediction unit (H_{pu}) images.

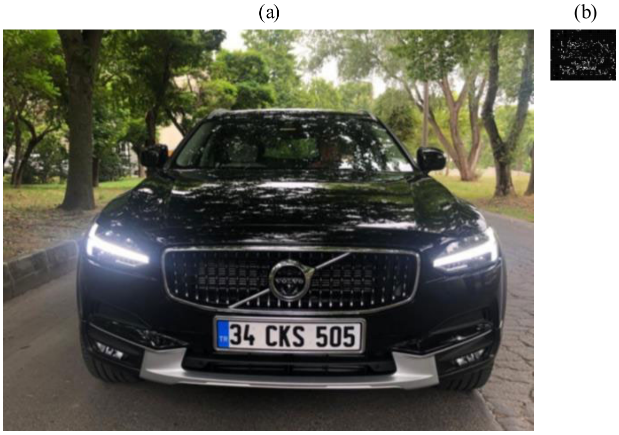

FIGURE 7. The size of the compressed domain image in comparison to the size of the pixel domain image. a) A 1024 × 768 pixel domain image. b) A compressed domain image corresponding to the pixel domain image with a resolution of 128 × 96 pixels, where each CU is represented by a single pixel.

pixel domain image that is decoded in accordance with the HEVC standard. The other is HEVC domain images, which are generated using our methods. The HEVC image generated can be H_{bp} or H_{pu}.

To generate a pixel domain image from a HEVC stream, the following steps are required: entropy decoding, dequantization, data inverse transformation, summing inverse transformed data with inter predicted data, applying a deblocking filter, and finally applying an adaptive loop filter. The required data for creating HEVC images, on the other hand, is available immediately after the entropy decoder step. There is no need for de-quantization or inverse transformation. According to [6] entropy decoding accounts for 37% of the total decoding phase. When generating HEVC images, we spend less computation time than when generating pixel domain images.

We have three distinct image types, and for each of these types, we generate unique YOLOv3-tiny weights. Weights for the pixel domain are obtained by training pixel domain images, whereas weights for the HEVC domain are obtained by training HEVC domain images. Images are fed into the YOLO network with the appropriate weight, and LP regions are detected. LPD_{pixel} is the abbreviation for the entire LP detector method in the pixel domain. There are two LP detectors in the compressed domain, and they are referred to as LPD_{CD_bp} and LPD_{CD_pu}, respectively, based on their input image types (cf. Fig. 8).

V. DATASETS

Despite the widespread use of LPR systems around the world, there are far too few datasets that are open to the public. This paper introduces and shares a new public-domain dataset, the Compressed Domain LP Dataset (CP-LP Dataset) [34]. Furthermore, a second data set, EnglishLP [35], is employed to compare the results with previously conducted studies.

A. CD-LP DATASET

The CD-LP dataset contains images from commercial cameras that are currently operational and located on a highway and at a shopping mall's entrance [36]. Generally, the images in this dataset depict the front view of a vehicle. They typically contain a single vehicle, but can occasionally contain two or more. The dataset contains 3 × 2.400 images in three formats: 2,400 *P*, 2,400 H_{bp} and 2,400 H_{pu}. H_{bp} and H_{pu} are created by first encoding pixel domain images into a HEVC bitstream and then utilizing one of the methods described in Section 4.

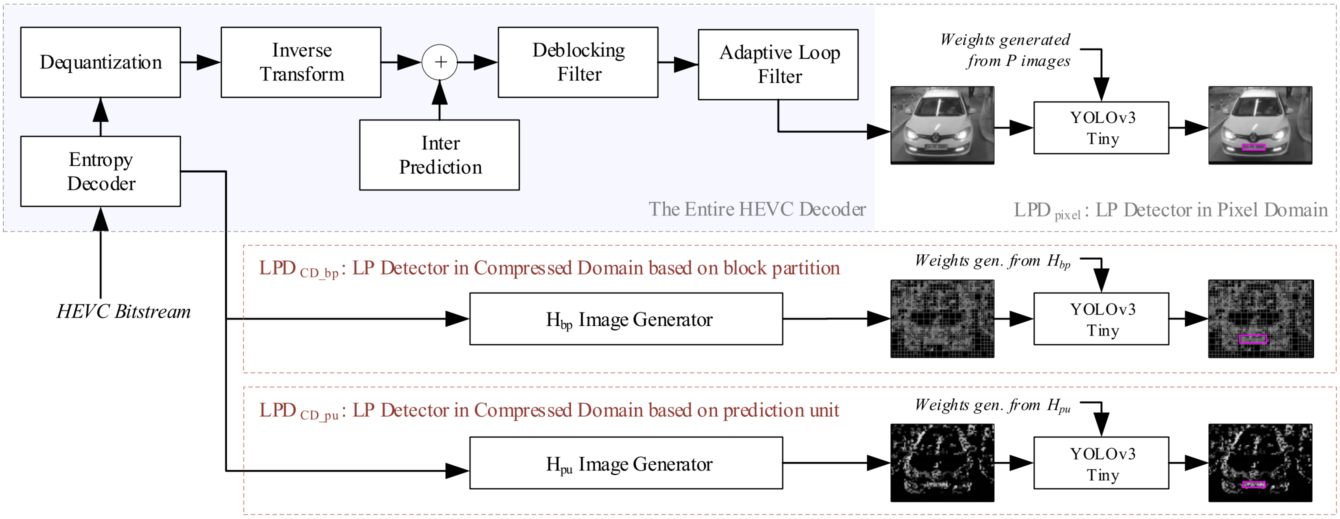

FIGURE 8. The process of generating HEVC images and detecting LPs for three distinct methods.

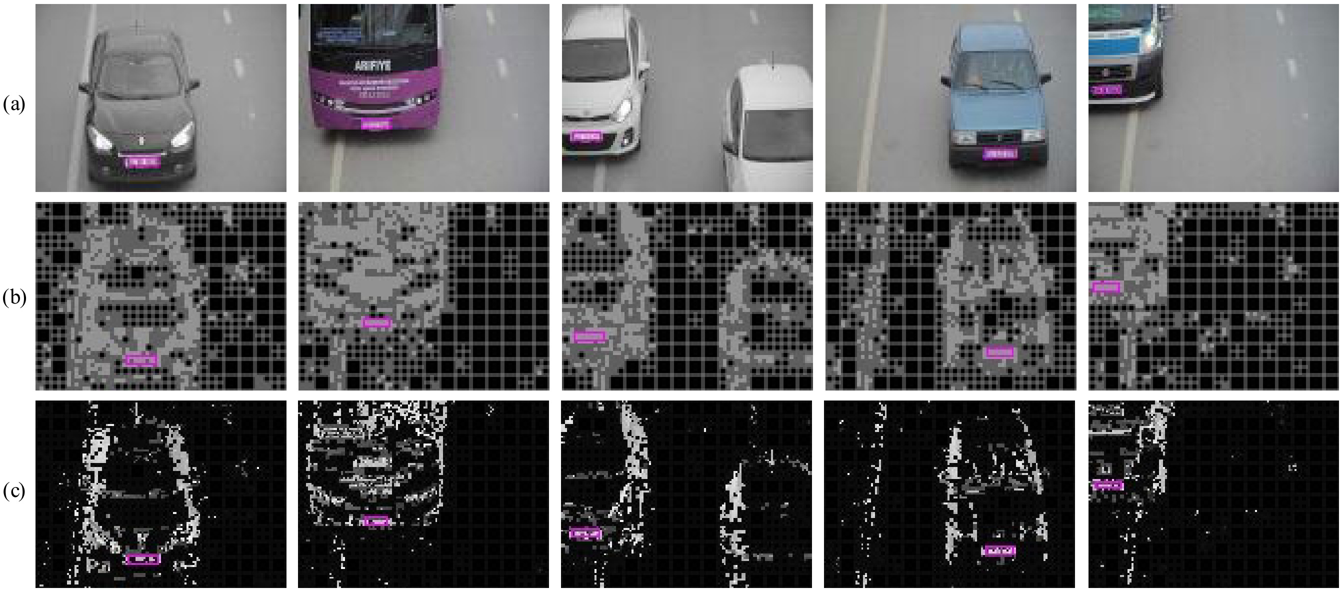

FIGURE 9. LP detection results from the test database for three distinct methods. A pink rectangle indicates plates that have been detected. a) Detection results of LPD_{pixel}. b) Detection results of LPD_{CD_bp}. c) Detection results of LPD_{CD_pu}.

The original images had a resolution of 1,024 × 768 pixels in size. Due to fact that each CU is represented by a single pixel, the H_{bp} and H_{pu} are generated at a resolution of 128×96 pixels. We also resize the P set to 128 × 96 pixels for three reasons:

1. A reasonable comparison to HEVC images.
2. To be able to make a database publicly accessible without regard for privacy concerns.
3. It has been demonstrated that a resolution of 128 × 96 pixels is sufficient for achieving high accuracy in LP detection.

The train set includes 1,800 images for each of the three formats, while the test set contains the remaining 600. Our dataset is summarized in Table 1. Each image in the database has a companion file containing plate annotation information in YOLO format.

TABLE 1. CD-LP dataset.

Image	Resolution	Train Set	Test Set	Total
P	1024×768	1,800	600	2,400
H_{bp}	128×96	1,800	600	2,400
H_{pu}	128×96	1,800	600	2,400

B. EnglishLP DATASET

A digital camera with a resolution of (640 × 480) pixels was used to capture images of the EnglishLP dataset. Over 500 images of the rear views of various vehicles (trucks, cars, buses) were included in the database, taken under various lighting conditions (cloudy, sunny, rainy).

This dataset is divided in the same way as in [16] and [37], with 80% of the images being used for training. %20 of the images are used for testing as given in Table 2.

TABLE 2. EnglishLP dataset.

Image	Resolution	Train Set	Test Set	Total
P	640×480	407	102	509
H_{bp}	128×96	407	102	509
H_{pu}	128×96	407	102	509

VI. EXPERIMENTAL RESULTS

A. MEASUREMENT

The following conditions were used to evaluate LP detection in the compressed and pixel domains:

1. The same number of images with the same resolution were used for training and testing.

2. The same deep learning network was utilized using the same hyper-parameters, including "learning rate", "batch size", "number of epochs to train for" and "number of nodes in the given layer".

3. A total of 500,000 training sessions are conducted to ensure that the average loss no longer decreases, and the weight with the best mAP is chosen from among the generated weights.

The results are evaluated using the F1-score, precision, recall, average intersection of union (Avg. IoU) and mean average precision (mAP). The Precision (P), Recall(R) and F1-score ($F1$) values are calculated based on True Positive (TP), False Positive (FP) and False Negative (FN) as shown in (3), (4) and (5), respectively. The F1-score is a metric that evaluates the sensitivity and accuracy criteria together.

$$R = \frac{TP}{TP + FN} \tag{3}$$

$$P = \frac{TP}{TP + FP} \tag{4}$$

$$F1 = \frac{2 \times P \times R}{P + R} \tag{5}$$

B. COMPARISON OF DATA SIZE

Analyzing data in the compressed domain allows us to work with less data. A comparison of data amount for each image type is shown in Table 3. In the pixel domain, the raw image is a three-channel colored image with a resolution of 1,024 × 768. H_{pu}, on the other hand, is a grayscale image with a 128 × 96 resolution. It contains 192 times less data than a pixel image. Finally, H_{bp} is a binary image with a resolution of 128 × 96 pixels. It is 1,536 times smaller than the pixel image.

C. ACCURACY

1) CD-LP DATASET ACCURACY

LP detection results for three different methods are given in Table 4. Pixel domain has the highest mAP overall. For the first HEVC domain method based on block partition, the mAP result is 82.78%. Although it has 1,536 times less amount of data, the result shows that detection of a plate is possible by this method in an extend. On the other hand, for the second HEVC domain method based on prediction units, achieved performance is very promising. The mAP accuracy is close to pixel domain while it can do it with 192 times less amount of data. The difference is only 1.73%.

TABLE 3. Data size comparison.

Image	Size	Depth	Size in bit	Ratio
P	1,024×768	24 (color)	18,874,368 bit	1/1
H_{pu}	128×96	8 (gray)	98,304 bit	1/192
H_{bp}	128×96	1 (B&W)	12,288 bit	1/1,536

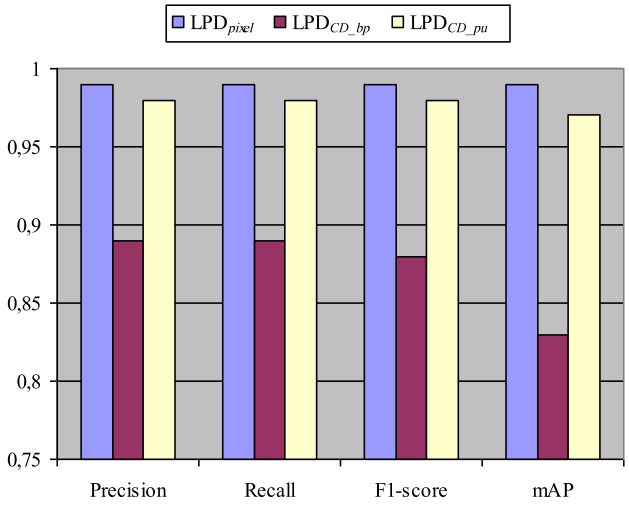

FIGURE 10. The results of LP detection for the CD-LP dataset are presented. A compressed domain approach based on prediction units achieves performance that is comparable to that of the pixel domain.

TABLE 4. CD-LP dataset accuracy.

Method	TP	FP	P	R	F1	mAP
LPD_{pixel}	598	4	0.99	0.99	0.99	99.11
LDP_{CD_bp}	533	70	0.88	0.89	0.88	82.78
LDP_{CD_pu}	588	11	0.98	0.98	0.98	97.38

Detection results, where "TP" means true positive, "FP" means false positive, "P" means precision and "R" means recall.

2) EnglishLP DATASET ACCURACY

EnglishLP is a publicly available dataset that enables us to compare methods developed in the compressed domain to those developed in the pixel domain. The obtained results are summarized in Table 5. The block partitioning method achieved a recall rate of 0.94. The partition unit-based method achieved 1.00 recall and precision rates by correctly detecting all LPs in the test set. This is the same rate as the Pixel domain approach and the research published in [17]. Despite being in the pixel domain, the study in [37] was unable to identify some LPs. The applicability of the proposed method appears to be promising considering these findings. The proposed

TABLE 5. EnglishLP dataset accuracy.

Method	TP	FP	P	R	F1	mAP
LPD_{pixel}	102	0	1.00	1.00	1.00	99.02%
LDP_{CD_bp}	96	6	0.94	0.94	0.94	93.33%
LDP_{CD_pu}	102	0	1.00	1.00	1.00	99.02%
[37]	-	-	-	0.98	-	-
[16]	-	-	-	1.00	-	-

Detection results, where "TP" means true positive, "FP" means false positive, "P" means precision and "R" means recall. The dash indicates that no data is shared for the related measurement.

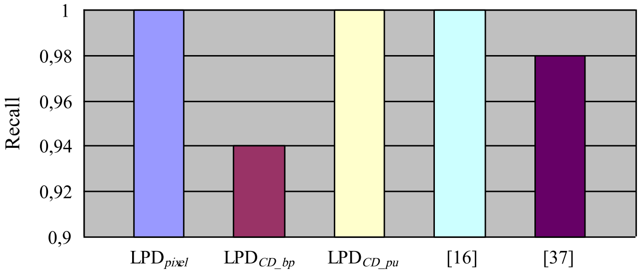

FIGURE 11. The LP detection results for the publicly accessible English-LP dataset.

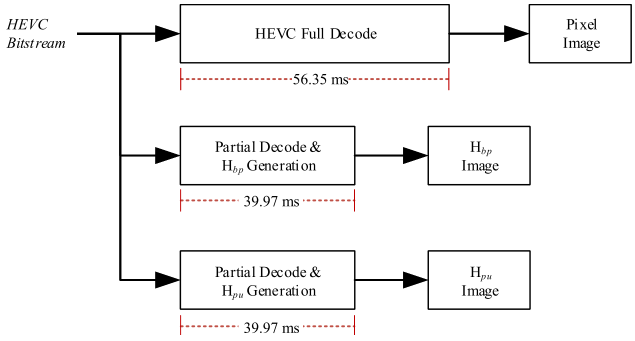

FIGURE 12. The time required to generate the image. Image generation in the compressed domain is 40% faster than the pixel domain.

LDP_{CD_pu} method successfully detects all plates while using 192 times less data than the pixel domain approach.

D. SPEED

One of the most important advantages of LP detection in HEVC domain is the reduction of the processes performed and the acceleration of the analysis.

1) IMAGE GENERATION

The required data for HEVC domain images is available at the end of the entropy decoder step. The steps of de-quantization and inverse transformation are omitted. However, for pixel images, the entire decoding phase must be completed. According to [6] entropy decoding accounts for 37% of the total decoding phase.

Using reference software [7], we measure the elapsed time during the decoding and image generation phases, as shown in Fig. 6. The process took an average of 85 milliseconds to generate an image in the pixel domain. The process of generating compressed domain images, on the other hand, took an average of 53 milliseconds. Our proposed method is more than 1.5 times faster than the traditional full decoding approach, as shown in Table 6.

TABLE 6. Time comparison of pixel and compressed domain image generation.

Time (ms)	LPD_{pixel}	LPD_{CD_bp}	LPD_{CD_pu}
Average	56.35	39.97	39.97
Minimum	36	23	23
Maximum	80	62	62

The experiments are done on an Intel(R) Core(TM) i9-9900X CPU @ 3.50GHz CPU, NVIDIA GeForce GTX 1080 Ti GPU, 48 GB RAM and Windows10 64-bit operating system using JCT-VC HEVC reference software version HM 16.9.

2) LP DETECTION

After the image has been created, the next step is to locate the license plate. The image produced in the compressed domain has been shown to have an 8 × 8 lower resolution than the image produced in the pixel domain. This low resolution significantly reduces the time required for the trained deep learning network to process images. As given in Table 7 these images are processed in about two milliseconds. This corresponds to a frame rate of 500 frames per second, which is more than enough for real time processing.

TABLE 7. Processing time for LP detection using YOLOv3 Tiny at various image resolutions.

Resolution	Time (ms)	FPS
128 × 96	2	500
416 × 416	3.6	280

The experiments are done on an Intel(R) Core(TM) i9-9900X CPU @ 3.50GHz CPU, NVIDIA GeForce GTX 1080 Ti GPU, 48 GB RAM and Windows10 64-bit operating system.

The general approach in the pixel domain is to use one of the default YOLO resolutions. When a 416 × 416 is used as network input, the processing time is 3.6 ms on average. Images generated in the pixel area can be processed at the same rate as images in the compressed domain. To accomplish this, the generated image must first be down scaled 8 × 8, and then the DNN must be trained with this resolution in mind.

3) OVERALL ACCELERATION

The entire process consists of image generation and LP detection. Comparing Table 6 and 7, it is apparent that image generation is the most time-consuming process, taking roughly 20 to 30 times the amount of time as LP detection. The proposed method suggests a significant improvement for this time-consuming phase. When the entire process is taken into account, the acceleration remains greater than 1.4 times (cf. Table 8).

VII. DISCUSSION AND FUTURE WORK

Within the scope of this study, a solution is represent only for LP detection part. The future work is combining this

TABLE 8. Comparison of the duration of the entire process.

Time (ms)	LPD_{pixel}	LPD_{CD_bp}	LPD_{CD_pu}
Average	58.35	41.97	41.97

All methods are compared at a 128×96 image resolution.

study with LP recognition. There are some advantages to do LP recognition in compressed domain. HEVC image format allows partial decoding of an image [5]. Thus, only a specific region of an image can be decompressed independent from the entire frame. That feature can be effectively used by our method since we can detect plate region inside whole encoded stream. Once plate region is detected, LP recognition can be performed at higher speeds by simply dissolving the relevant area.

Having the ability to identify LPs in the compressed domain would pave the way for advanced codecs and applications in video analytics. The privacy-protected cameras that display cars while shielding LP areas could be one such potential application.

Another significant application of the study could be the scanning of video archives. Vehicle and license plate searches can be performed on compressed videos by leveraging the method's speed advantage.

VIII. CONCLUSION

To the best of our knowledge, there have been no previous studies on LP detection in compressed video streams. Our research demonstrates that it is possible to detect vehicle plates in a compressed domain. The study expands LP detection into a new area, which has promising results in terms of speed and accuracy. Compressed domain analytics has a big potential in solving performance bottleneck for common artificial intelligence tasks. Another important contribution is that we also share our database to assist future research on compressed domain LP detection.

REFERENCES

[1] M. S. Beratoglu and B. U. Toreyin, "Vehicle license plate detection using only block partitioning structure of the high efficiency video coding (HEVC)," in *Proc. 27th Signal Process. Commun. Appl. Conf. (SIU)*, Apr. 2019, pp. 1–4, doi: 10.1109/SIU.2019.8806429.

[2] A. Bhujbal and D. Mane, "A survey on deep learning approaches for vehicle and number plate detection," *Int. J. Sci. Technol. Res.*, vol. 8, pp. 1378–1383, Oct. 2020.

[3] W. Weihong and T. Jiaoyang, "Research on license plate recognition algorithms based on deep learning in complex environment," *IEEE Access*, vol. 8, pp. 91661–91675, 2020, doi: 10.1109/access.2020.2994287.

[4] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560–576, Jul. 2003, doi: 10.1109/TCSVT.2003.815165.

[5] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012, doi: 10.1109/TCSVT.2012.2221191.

[6] F. Bossen, *Common HM Test Conditions and Software Reference Configurations*, document JCTVC-L1100 ISO/IEC JTC1/SC29 WG11 m28412, 2013.

[7] *JCT-VC HEVC Reference Software Version HM 16.9.* Accessed: Jun. 28, 2021. [Online]. Available: https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/tags/HM-16.9/

[8] N. Farajian and M. Rahimi, "Algorithms for licenseplate detection: A survey," in *Proc. Int. Congr. Technol., Commun. Knowl.*, 2015, pp. 1–8, doi: 10.1109/ICTCK.2014.7033529.

[9] D. Zheng, Y. Zhao, and J. Wang, "An efficient method of license plate location," *Pattern Recognit. Lett.*, vol. 26, no. 15, pp. 2431–2438, Nov. 2005, doi: 10.1016/j.patrec.2005.04.014.

[10] B. B. Yousif, M. M. Ata, N. Fawzy, and M. Obaya, "Toward an optimized neutrosophic K-means with genetic algorithm for automatic vehicle license plate recognition (ONKM-AVLPR)," *IEEE Access*, vol. 8, pp. 49285–49312, 2020, doi: 10.1109/ACCESS.2020.2979185.

[11] M. Zahedi and S. M. Salehi, "License plate recognition system based on SIFT features," *Procedia Comput. Sci.*, vol. 3, pp. 998–1002, Jun. 2011, doi: 10.1016/j.procs.2010.12.164.

[12] F. Kahraman, B. Kurt, and M. N. D. Gökmen, "License plate character segmentation based on the Gabor transform and vector quantization," in *Proc. Int. Symp. Comput. Inf. Sci.*, Nov. 2003, pp. 381–388.

[13] I. V. Pustokhina, D. A. Pustokhin, J. J. P. C. Rodrigues, D. Gupta, A. Khanna, K. Shankar, and G. P. Joshi, "Automatic vehicle license plate recognition using optimal K-means with convolutional neural network for intelligent transportation systems," *IEEE Access*, vol. 8, pp. 92907–92917, 2020, doi: 10.1109/access.2020.2993008.

[14] F. Delmar Kurpiel, R. Minetto, and B. T. Nassu, "Convolutional neural networks for license plate detection in images," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2017, pp. 3395–3399, doi: 10.1109/ICIP.2017.8296912.

[15] Hendry and R.-C. Chen, "Automatic license plate recognition Via sliding-window darknet-YOLO deep learning," *Image Vis. Comput.*, vol. 87, pp. 47–56, Jul. 2019, doi: 10.1016/j.imavis.2019.04.007.

[16] W. Wang, J. Yang, M. Chen, and P. Wang, "A light CNN for end-to-end car license plates detection and recognition," *IEEE Access*, vol. 7, pp. 173875–173883, 2019, doi: 10.1109/ACCESS.2019.2956357.

[17] R. Laroca, L. A. Zanlorensi, G. R. Gonçalves, E. Todt, W. R. Schwartz, and D. Menotti, "An efficient and layout-independent automatic license plate recognition system based on the YOLO detector," *IET Intell. Transp. Syst.*, vol. 15, pp. 483–503, Apr. 2021, doi: 10.1049/itr2.12030.

[18] W. Min, X. Li, Q. Wang, Q. Zeng, and Y. Liao, "New approach to vehicle license plate location based on new model YOLO-L and plate pre-identification," *IET Image Process.*, vol. 13, no. 7, pp. 1041–1049, May 2019, doi: 10.1049/iet-ipr.2018.6449.

[19] J. Tao, W. Hu, and J. Ouyang, "Research and implementation of license plate location based on improved Yolo algorithm," in *Proc. 2nd Int. Conf. Inf. Technol. Electr. Eng.* New York, NY, USA: ACM, Dec. 2019, pp. 1–7, doi: 10.1145/3386415.3386967.

[20] S. M. Silva and C. R. Jung, "License plate detection and recognition in unconstrained scenarios," in *Computer Vision—ECCV* (Lecture Notes in Computer Science), vol. 11216, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds. Cham, Switzerland: Springer, 2018, doi: 10.1007/978-3-030-01258-8_36.

[21] R. V. Babu, M. Tom, and P. Wadekar, "A survey on compressed domain video analysis techniques," *Multimedia Tools Appl.*, vol. 75, no. 2, pp. 1043–1078, Jan. 2016, doi: 10.1007/s11042-014-2345-z.

[22] M. Javed, P. Nagabhushan, and B. B. Chaudhuri, "A review on document image analysis techniques directly in the compressed domain," *Artif. Intell. Rev.*, vol. 50, no. 4, pp. 539–568, 2017, doi: 10.1007/s10462-017-9551-9.

[23] R. Torfason, F. Mentzer, E. Agustsson, M. Tschannen, R. Timofte, and L. Van Gool, "Towards image understanding from deep compression without decoding," in *Proc. 6th Int. Conf. Learn. Represent.*, 2018, pp. 1–17. [Online]. Available: http://arxiv.org/abs/1803.06131

[24] B. U. Töreyin, A. E. Cetin, A. Aksay, and M. B. Akhan, "Moving region detection in compressed video," in *Computer and Information Sciences* (Lecture Notes in Computer Science), vol. 3280, C. Aykanat, T. Dayar, and I. Körpeoglu, Eds. Berlin, Germany: Springer, 2004, pp. 381–390.

[25] M. M. Y. Savci Yildirim, G. Saygili, and B. U. Töreyin, "Fire detection in H.264 video," in *Proc. ICASSP*, Brighton, U.K., May 2019, pp. 12–17.

[26] A. Benazza-Benyahia, N. Hamouda, F. Tlili, and S. Ouerghi, "Early smoke detection in forest areas from DCT based compressed video," in *Proc. 20th Eur. Signal Process. Conf. (EUSIPCO)*, Bucharest, Romania, 2012, pp. 2752–2756.

[27] B. Töreyin, "Smoke detection in compressed video," *Proc. SPIE Appl. Digit. Image Process.*, vol. 10752, Sep. 2018, Art. no. 1075232.

[28] F. Bombardelli, S. Gul, D. Becker, M. Schmidt, and C. Hellge, "Efficient object tracking in compressed video streams with graph cuts," in *Proc. IEEE 20th Int. Workshop Multimedia Signal Process. (MMSP)*, Feb. 2018, pp. 1–6, doi: 10.1109/MMSP.2018.8547120.

[29] L. Zhao, Z. He, W. Cao, and D. Zhao, "Real-time moving object segmentation and classification from Hevc compressed surveillance video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 6, pp. 1346–1357, Jun. 2018, doi: 10.1109/TCSVT.2016.2645616.

[30] S. R. Alvar, H. Choi, and I. V. Bajic, "Can you find a face in a Hevc bitstream?" in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Calgary, AB, Canada, Apr. 2018, pp. 1288–1292, doi: 10.1109/ICASSP.2018.8462654.

[31] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 779–788, doi: 10.1109/CVPR.2016.91.

[32] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*. [Online]. Available: http://arxiv.org/abs/1804.02767

[33] *Darknet: Open Source Neural Networks*. Accessed: Jun. 16, 2020. [Online]. Available: https://pjreddie.com/darknet/

[34] Muhammet Sebul Beratoglu, Behcet Ugur Toreyin. *CD-LP: Compressed Domain License Plate Detection Database*. Accessed: Aug. 24, 2020. [Online]. Available: https://dx.doi.org/10.21227/2511-ym76

[35] V. Srebric. (2003). *EnglishLP Database*. [Online]. Available: http://www.zemris.fer.hr/projects/LicensePlates/english/baza_slika.zip

[36] *Divit Technology*. Accessed: Jun. 16, 2020. [Online]. Available: http://www.divit.com.tr/

[37] S. Azam and M. M. Islam, "Automatic license plate detection in hazardous condition," *J. Vis. Commun. Image Represent.*, vol. 36, pp. 172–186, Apr. 2016, doi: 10.1016/j.jvcir.2016.01.015.

[38] J. Ohm, G. J. Sullivan, H. Schwarz, T. K. Tan, and T. Wiegand, "Comparison of the coding efficiency of video Coding standards-including high efficiency video coding (HEVC)," in *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1669–1684, Dec. 2012, doi: 10.1109/TCSVT.2012.2221192.

MUHAMMET SEBUL BERATOĞLU received the B.S. degree in control and computer engineering and the M.S. degree in computer engineering from İstanbul Technical University, İstanbul, Turkey, in 2000 and 2003, respectively, where he is currently pursuing the Ph.D. degree in computer sciences with the Informatics Institute. His research interests include signal processing and pattern recognition, particularly as they relate to smart cities, intelligent transportation systems, and the Internet of Things.

BEHÇET UĞUR TÖREYİN received the B.S. degree from the Middle East Technical University, Ankara, Turkey, in 2001, and the M.S. and Ph.D. degrees from Bilkent University, Ankara, in 2003 and 2009, respectively, all in electrical and electronics engineering. He is currently an Associate Professor with the Informatics Institute, İstanbul Technical University. His research interests include signal processing, pattern recognition with applications to computational intelligence, and developing novel algorithms to analyze and compress signals from multitude of sensors, such as visible/infra-red/hyperspectral cameras, microphones, passive infra-red sensors, vibration sensors, and spectrum sensors for wireless communications.