Fast Object Detection in HEVC Intra Compressed Domain

Liuhong Chen*, Heming Sun†‡, Jiro Katto†, Xiaoyang Zeng* and Yibo Fan*

*Fudan University, Shanghai, P.R. China; †Waseda University, Tokyo, Japan; ‡JST, PRESTO, Saitama, Japan

***Abstract*—Conventional object detection methods are in the pixel domain and require full decoding with high computational complexity. In this paper, we propose a fast object detection method in the intra compressed domain of High Efficiency Video Coding (HEVC), which significantly accelerates the object detection process that uses compressed video images. Considering the characteristics of various coding features, we select 3 types of data for object detection, including partitioning depths, prediction modes, and residuals. To achieve a more discriminative representation of the residuals, we design an iterative restoration algorithm that can generate the details of the original image and reduce the noise in the residuals. Extensive evaluations on multiple HEVC test sequences and large-scale object detection dataset BDD100K confirm the effectiveness of our method. With a slight reduction in detection accuracy, our compressed domain detection system runs 1.8 times faster than the pixel domain.**

***Index Terms*—Compressed domain video analysis, object detection, HEVC, intra frame**

I. Introduction

Video analysis is an important subject in the field of computer vision. Conventional video analysis methods use fully decoded images to extract features, which is also called pixel domain video analysis. The required decoding steps in the pixel domain are illustrated in Fig. 1, including entropy decoding, inverse quantization (IQ), inverse transform (IT), and intra reconstruction for I-frame or inter reconstruction for P-frame. Pixel domain analysis methods have two major disadvantages: 1) The computational cost of decoding is expensive; 2) The total decoding time is long, especially for the I-frame. To tackle these issues, compressed domain analysis methods have been proposed, using the data extracted with partial decoding. As shown in Fig. 1, there is no reconstruction step in compressed domain decoding. Many computer vision tasks have been explored with compressed domain analysis techniques, such as action recognition [1], [2], anomaly detection [3], object detection [4]–[15], object tracking [16], [17], etc. We only discuss the object detection task in this paper.

According to the targeted compression standard, existing compressed domain object detection methods are grouped into 3 classes: MPEG-based ([4], [5]), H.264-based ([6]–[8], [13]), and HEVC-based ([9]–[11], [14], [15]). In [4], coarse object detection was first completed in the compressed domain using Discrete Cosine Transform (DCT) coefficients and finer detection of the object edge was then achieved in the pixel domain with local inverse transform. Yokoyama *et al.* [5]

Corresponding author: Heming Sun, hemingsun@aoni.waseda.jp.

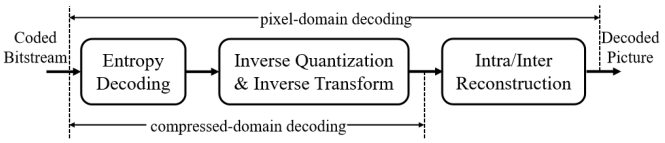


Fig. 1. Decoding steps of the pixel domain and the compressed domain.

utilized motion cues represented with Motion Vectors (MVs) to segment moving regions by classifying each coded block into a moving or non-moving one. More types of compressed-domain data were exploited in H.264-based methods, *e.g.*, the block size, the coded mode, and the quantization parameter (QP) of each coded block. It was common in [6]–[8] that MVs played the key role in object detection. Since the data used by these methods only had the block-level resolution, their detection results could only achieve block-level precision. Meanwhile, the generalized object detection shrank into moving object detection. Completely differently, Wang *et al.* [13] proposed to propagate the features of the pixel-domain I-frame into the features of the following P-frames within the same group of picture (GOP) based on MVs, which saved the feature extraction time of P-frames. Compared with H.264, HEVC introduced a more complex division method of coding blocks and more prediction modes in intrapicture prediction, which provided richer syntax elements for compressed-domain object detection. Zhao *et al.* [9], [11] first trained a foreground-background classifier to segment moving regions with MVs, Coding Unit (CU) sizes, Prediction Unit (PU) sizes, and coding modes (intra, inter or skip). They also designed a second classifier to predict the actual class label (vehicle or person) of the segmented regions. Alizadeh *et al.* [15] used a Conditional Random Field (CRF) model to detect moving regions, and their utilized data were assembled by MVs, CU partitioning modes, and the number of consumed bits.

In summary, the existing methods that only use compressed domain data have some disadvantages. First, since the data they use only has the block-level resolution, they cannot achieve pixel-level localization of objects. Secondly, excessive reliance on MV will impair detection accuracy. In fact, MVs are influenced by the target compression rate during encoding, and they deviate from the actual motion information. Besides, when the block in the P frame is coded with the intra type, the MV disappears, which usually occurs in the discriminative area of the object. Third, there is no available detection

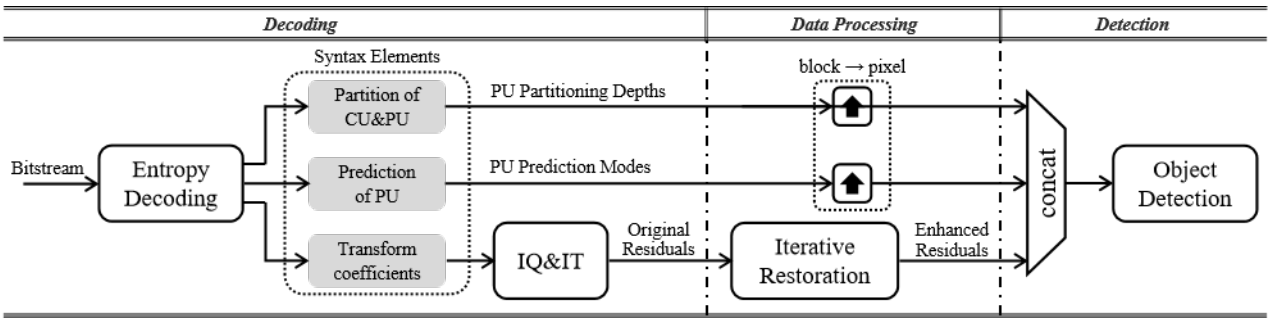


Fig. 2. The architecture of the proposed fast object detection method of using I-frame in the HEVC compressed domain.

method that only uses the compressed domain data of the I-frame. In practical applications, some videos are coded with the $intra\_main$ configuration in which all frames are intra-encoded. For these videos, the P-frame-based method becomes useless, and the I-frame-based method is required.

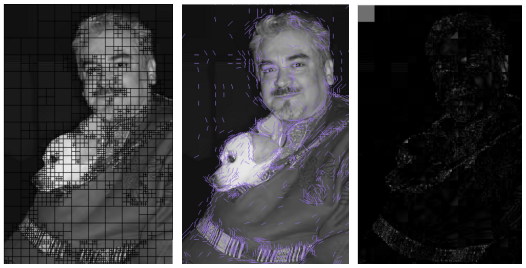
In this paper, we propose a novel object detection method in the HEVC intra compressed domain, which only requires one compressed I-frame as input. It can be easily integrated with existing P-frame-based methods to build a complete compressed domain object detection system. The data extracted from the compressed I-frame include the partitioning depths, the prediction modes, and the residuals. Since the residuals have the pixel-level resolution, our detection method achieves pixel-level object localization. Furthermore, we propose an iterative restoration algorithm to enhance the representation of the residuals. Finally, we extensively evaluate our fast object detection method on multiple HEVC test sequences and the object detection dataset BDD100K [18], and demonstrate that it can significantly accelerate the object detection procedure.

II. Proposed Methods

A. Compressed domain Object detection using HEVC I-frame

The overall framework of the proposed I-frame-based object detection method in the HEVC compressed domain is illustrated in Fig. 2. Only entropy decoding, IQ, and IT are required for decoding. The syntax elements obtained by entropy decoding contain the division and prediction information of all PUs, which determine the partitioning depth and prediction mode. Each PU has one partitioning depth and one prediction mode, and we assign the same given value to all pixels in one PU for up-sampling, which upgrades the block-level data into pixel-level. After IQ and IT, the pixel-level intrapicture prediction residuals are obtained and then restored through multiple iterations. Finally, all data extracted from the compressed I-frame are concatenated in the channel dimension. We choose the fast one-stage object detector [19] for object detection. Unlike [17] which also uses YOLO as an object detector in the compressed domain, we use compressed domain data as input while they employ the pixel domain RGB image.

During splitting coding tree block (CTB) into coding block (CB) and CB into prediction block (PB), the number of divisions increases as the local image contents get more complex, as shown in Fig. 3 (a). We regard the division times of CTB into PB as the partitioning depth of the PU, which is calculated as follows:



(a) Partition. (b) Prediction. (c) Residuals.

Fig. 3. Visualizations of data extracted from the HEVC I-frame.

$$Partition(x, y) = \log_2(size(CTB)/size(PB(x, y))) \quad (1)$$

$Partition$ represents the partitioning depth of PB. x and y are 2 position indexes for each pixel. Consistent with the HEVC $intra\_main$ configuration, the size L×L of a luma CTB is chosen as L=64 and the minimum size of CB is 8 (only the luma component is used in our proposal.). In (1), the $size(CTB)$ is equal to 64 and the value of $size(PB(x, y))$ is within $\{64, 32, 16, 8, 4\}$. $Partition$ weighs the significance of each PU because it reflects the information entropy of them.

There are a total of 35 modes supported in HEVC intrapicture prediction, which are $Planar$, DC, and 33 $Angular_k$. We use the mode number defined in HEVC to represent the prediction mode. The $Planar$ mode is applied to areas with smooth changes while the DC mode is for flat areas. The $Angular_k$ prediction method sets 33 prediction directions to achieve accurate prediction. Fig. 3 (b) visualizes the directionality of the $Angular_k$ prediction modes. It depicts that the directional orientation of the prediction mode and the changing direction of image content are greatly aligned.

In the coded bit-stream, the residual part occupies a huge volume and retains a fine description of the original image. As the prediction errors at the boundary areas are large, the residuals save the edge information of the object, as shown in Fig. 3 (c). To exploit more details of the original image, the residuals are used as reference to perform image prediction, and the predicted image is merged with the original residuals.

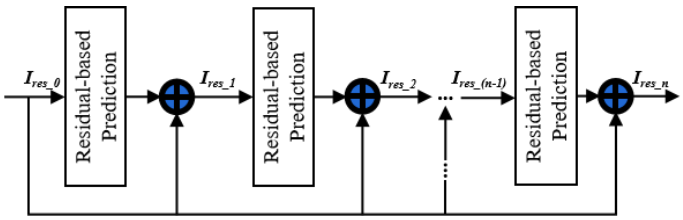


Fig. 4. The proposed iterative restoration with n iterations.

We call this operation the restoration of residuals. Compared with the original residual image, patterns in the restored image are more complete and clear. It motivates us to use the restored image as reference to restore, that is, iterative restoration.

B. The Iterative Restoration Based on Residuals

Patterns in the original residual image are incomplete due to the removal of spatial redundancy during encoding, and the representation extracted from residuals is too weak for robust object detection. To tackle this issue, we conduct restoration operation with multiple iterations as demonstrated in Fig. 4. The proposed iterative restoration is also functioned below:

$$I_{res}(t+1) = \begin{cases} F(I_{res}(0)) + I_{res}(0), & t = 0 \\ F(I_{res}(t)) + I_{res}(0), & t > 0 \ and \ t \in \mathbb{N} \end{cases} \quad (2)$$

$I_{res}(t)$ is the restored image at iteration t and $I_{res}(0)$ is the original residual image obtained after IT processing. F represents the residual-based image prediction algorithm. At each iteration, the original residual image will be merged into the predicted image. The restored image can be regarded as the enhanced version of residual image.

Compared with the predicted image or the original residual image alone, the restored image aggregated by the two is more reliable, providing more complete contours of the object and reducing the noise in the residual image. Then, we use the restored image as reference to perform image prediction again to acquire more details of the original frame image. It should be noted that the number of iteration is limited, and excessive iteration will degrade the quality of the restored image. Fig. 5 visualizes the restored images under different iterations. When the iteration number increases, both the subjective quality and the objective quality indicated by Peak Signal to Noise Ratio (PSNR) and Structural SIMilarity (SSIM) are enhanced.

Image prediction based on residuals acts as a generator of image details, making the discontinuous edges of objects in the original residual image continuous. The spatial continuity of the residual image is used in the prediction algorithm. As shown in Fig. 3 (c), the edge information of the original image is retained in the residual, indicating the texture or contour of the object. For the missing edge part, we believe that it can be inferred from the edge part available in the residual. Through residual-based image prediction, missing edges can be added and existing edges can be enhanced.

The predicted value of each pixel is a weighted average of selected reference samples. We fix the location of reference samples with a mimic of the intrapicture prediction procedure defined in HEVC. According to the position, division boundary, and prediction mode of transform units (TUs), a mask indicating the projected location for each pixel is obtained, as well as the weights required for the calculation of the predicted value. The prediction algorithm F is decomposed as follows:

$$R(x, y; t) = I_{res}(t-1)(M(x, y)) \quad (3)$$

$$I_{pred}(x, y; t) = \sum_{n=1}^{num(W(x,y))} W(x, y)[n] * R(x, y; t)[n] \quad (4)$$

$R(t)$ represents the reference samples required for each pixel in the t_{th} image prediction, and is composed of the $(t-1)_{th}$ restored residuals. $I_{pred}(t)$ is the predicted image of iteration t. W and M are weights and the projection mask, and they are only related to the divisions and prediction modes of TUs. x and y are 2 position indexes of each pixel. $W(x, y)$ and $M(x, y)$ are two lists, and their lengths are the same, which is equal to the number of reference samples of pixel (x, y).

Through (2), (3), and (4), it can be inferred that all pixels are processed simultaneously in each restoration. The proposed iterative reduction is a parallel operation, which can be accelerated by hardware. Therefore, the time cost of restoration is small compared with the time cost of decoding.

III. Experiments

A. Experimental Settings

Runtime. The decoding process is based on the HEVC reference software HM16.9 [20], and YOLO detector [19] is integrated into the HM project. Since video resolution and QP affect the decoding time, we conduct runtime experiment on multiple HEVC test sequences with 4 video resolutions {832x480, 1024x768, 1920x1080, 2560x1600} and 2 QPs {22, 32}. The acceleration on the object detection dataset BDD100K is also experimented with. Our heterogeneous system includes the Intel Xeon Gold 6134 CPU for decoding and the Nvidia GeForce RTX 2080 GPU for detection.

Accuracy. The large-scale dataset BDD100K [18] contains 10 classes of objects, of which 69, 863 are used for training and 10,000 for validation. All images are 1280x720 in size and stored in JPEG format. To convert them to HEVC compression standard, we decode all pictures with ffmpeg and then compress them with HEVC $intra\_main$ encoding. Hyperparameters used for training are consistent with the scratch in [19] and the training process is the same for all methods.

B. Runtime Analysis

Since related works [9]–[11], [14], [15] only provided experimental results for P-frames and our method only focus on the I-frame, it's hard to make an aligned comparison between us. So, we demonstrate the acceleration performance of our method by comparing the total runtime of the pixel domain and the compressed domain, as recorded in Table I. Below is the calculation method of the time saved:

$$T_{saved} = 1 - (T_{compressed} \ / \ T_{pixel}) \quad (5)$$

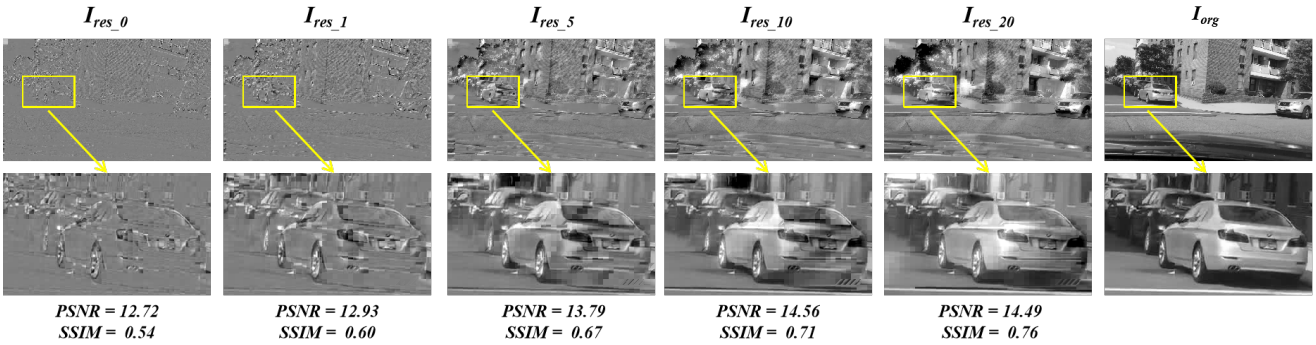


Fig. 5. Visualization of the restored images based on the proposed iterative restoration method. From left to right are the original residual image, 4 restored images and the original grayscale image. Quantified indicators of the image quality are attached below each image.

TABLE I
THE RUNTIME COMPARISON BETWEEN COMPRESSED DOMAIN AND PIXEL DOMAIN. THERE ARE 3 TYPES OF TIME MEASUREMENT RESULTS, LABELED AS $\{T_{decoding}, T_{detection}, T_{total}\}$. THE DATA PROCESSING TIME OF THE COMPRESSED DOMAIN IS INCLUDED IN ITS $T_{decoding}$ PART.

QP	Sequence	Resolution	Pixel domain(ms)			Compressed domain(ms)			Time Saved(%)	
			$T_{decoding}$	$T_{detection}$	T_{total}	$T_{decoding}$	$T_{detection}$	T_{total}	$T_{decoding}$	T_{total}
22	BasketballDrill	832x480	37.22	22.04	59.26	19.00	21.77	40.77	48.95	**31.20**
22	ChinaSpeed	1024x768	64.50	23.93	88.43	31.57	26.97	58.54	51.05	**33.80**
22	ParkScene	1920x1080	173.23	29.87	203.10	99.97	29.17	129.14	42.29	**36.42**
22	PeopleOnStreet	2560x1600	325.17	38.39	363.56	169.84	37.16	207.00	47.77	**43.06**
22	BDD100K	1280x720	55.65	33.18	88.83	28.08	28.31	56.39	49.54	**36.52**
32	BasketballDrill	832x480	24.66	21.14	45.80	10.06	20.12	30.18	59.21	**34.10**
32	ChinaSpeed	1024x768	50.17	24.98	75.15	22.27	26.27	48.54	55.61	**35.41**
32	ParkScene	1920x1080	120.58	31.93	152.51	55.52	30.90	86.42	53.96	**43.33**
32	PeopleOnStreet	2560x1600	250.55	40.85	291.40	109.80	36.39	146.19	56.18	**49.83**
32	BDD100K	1280x720	43.29	30.66	73.95	20.47	24.34	44.81	52.71	**39.41**

For compressed domain method, as we hope to obtain higher detection accuracy, the iteration number of restoration is set as 20. Besides, the detection in the compressed domain takes a little less time than the pixel domain, which may be that the compressed data contains more zeros.

For middle resolution 480P, the time saved is 31.2% (QP=22) and the acceleration is 1.45 times. For high-definition 1080P sequence, a 1.8x speedup is obtained (QP=32). When the resolution reaches 1600P, our method can achieve an acceleration of more than 1.8 times. This is because the acceleration is determined by the ratio of time spent on entropy decoding and intra reconstruction respectively. Considering the trade-off between the acceleration and the detection accuracy, the optimal QP is 22, as shown in Fig. 7. Runtime results in Table I demonstrate that the process of object detection has been significantly accelerated with our method.

C. Accuracy Analysis

Comparison with pixel domain. To prove the feasibility of our method, we test the detection accuracy on a large-scale object detection dataset BDD100K using the standard mean average precision (mAP) metric. Images in BDD100K are sampled from complex and realistic traffic scenes. Experimental results are provided in Table II. The method using RGB or grayscale image belongs to the pixel domain, and the others belong to the compressed domain (QP=22). The detection accuracy of the RGB image is the highest, and the original residual is the lowest. The difference between the two is 8.22% and 5.15%. After restoring the residuals 20 times, the accuracy is improved by 1.80% and 1.21% respectively, and the performance gap with RGB is reduced to 6.42% and 3.94%. Fig. 6 is the visualization of object detection in the pixel domain and compressed domain.

TABLE II
THE MAPS OF DIFFERENT METHODS ON THE DATASET BDD100K

Input	$mAP_{0.5}(\%)$	$mAP_{0.5:0.95}(\%)$
$I_{res(t=0)}$	48.90	26.47
$I_{res(t=1)}$	49.34	26.80
$I_{res(t=5)}$	50.07	27.16
$I_{res(t=10)}$	50.46	27.44
$I_{res(t=20)}$	50.70	27.68
I_{gray}	54.75	30.24
I_{rgb}	57.12	31.62

Quantization step length. With the increase of QP, the volume of residuals in the bitstream decreases, and the detection

(a) Detection of dense objects. (b) Better in compressed domain.

Fig. 6. Visualization of object detection. For each example, the pixel domain is on the left and the compressed domain is on the right.

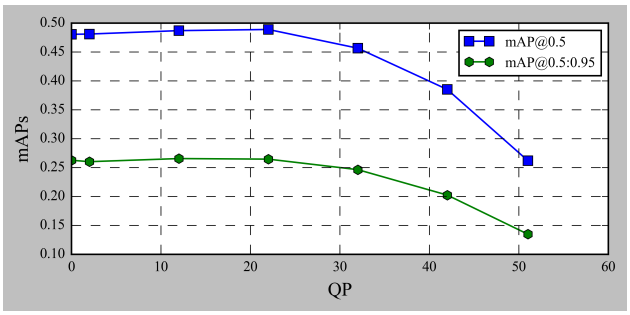


Fig. 7. The relationship between detection accuracy mAP and QP.

accuracy will be affected. The relationship between QP and detection accuracy mAP is depicted in Fig. 7. The accuracy first rises gently and then drops sharply. When QP is relatively high, the useful information contained in the residuals will be little. And more small-value noise will appear in the decoded residuals and degrade detection accuracy as QP is very low.

IV. CONCLUSION

In this paper, we propose a novel fast object detection method in the HEVC intra compressed domain. Multiple types of compressed data that help detection are exploited, including the partitioning depths, the prediction modes, and the residuals. We also design a fast restoration method to enhance the representation of residuals. This operation supports parallel processing and can be executed iteratively. Finally, we choose multiple HEVC test sequences and a large-scale object detection dataset BDD100K for convincing verification. Experimental results about runtime and detection accuracy confirm the effectiveness of our proposals. As I-frame could exist in any compressed video, our method can be integrated with other P-frame-based methods, and it only requires one independent I-frame. In the future, our research will be extended to the Versatile Video Coding (VVC) compression standard, and the detection method of using compressed P-frames will be included.

V. ACKNOWLEDGEMENTS

This work was supported in part by the National Natural Science Foundation of China under Grant 62031009, in part by the Shanghai Science and Technology Committee (STCSM) under Grant 19511104300, in part by Alibaba Innovative Research (AIR) Program, in part by the Innovation Program of Shanghai Municipal Education Commission, in part by the Fudan University-CIOMP Joint Fund (FC2019-001), in part by Kenjiro Takayanagi Foundation, in part by JST, PRESTO under Grant JPMJPR19M5.

REFERENCES

[1] C. Wu, M. Zaheer, H. Hu, R. Manmatha, A. J. Smola, and P. Krähenbühl, "Compressed video action recognition," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6026–6035.

[2] Z. Shou, X. Lin, Y. Kalantidis, L. Sevilla-Lara, M. Rohrbach, S. Chang, and Z. Yan, "DMC-Net: Generating discriminative motion cues for fast compressed video action recognition," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1268–1277.

[3] Y. Zhang and H. Chao, "Abnormal event detection in surveillance video: A compressed domain approach for HEVC," in *2017 Data Compression Conference (DCC)*, 2017, pp. 475–475.

[4] O. Sukmarg and K. R. Rao, "Fast object detection and segmentation in MPEG compressed domain," in *2000 TENCON Proceedings. Intelligent Systems and Technologies for the New Millennium (Cat. No.00CH37119)*, vol. 3, 2000, pp. 364–368 vol.3.

[5] T. Yokoyama, T. Iwasaki, and T. Watanabe, "Motion vector based moving object detection and tracking in the MPEG compressed domain," in *2009 Seventh International Workshop on Content-Based Multimedia Indexing*, 2009, pp. 201–206.

[6] S. K. Kapotas and A. N. Skodras, "Moving object detection in the H.264 compressed domain," in *2010 IEEE International Conference on Imaging Systems and Techniques*, 2010, pp. 325–328.

[7] E. Maekawa and S. Goto, "Examination of a tracking and detection method using compressed domain information," in *2013 Picture Coding Symposium (PCS)*, 2013, pp. 141–144.

[8] M. Laumer, P. Amon, A. Hutter, and A. Kaup, "Compressed domain moving object detection by spatio-temporal analysis of H.264/AVC syntax elements," in *2015 Picture Coding Symposium (PCS)*, 2015, pp. 282–286.

[9] L. Zhao, D. Zhao, X. Fan, and Z. He, "HEVC compressed domain moving object detection and classfication," in *2016 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2016, pp. 1990–1993.

[10] J. De Praeter, J. Van de Vyver, N. Van Kets, G. Van Wallendael, and S. Verstockt, "Moving object detection in the HEVC compressed domain for ultra-high-resolution interactive video," in *2017 IEEE International Conference on Consumer Electronics (ICCE)*, 2017, pp. 135–136.

[11] L. Zhao, Z. He, W. Cao, and D. Zhao, "Real-time moving object segmentation and classification from HEVC compressed surveillance video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 6, pp. 1346–1357, 2018.

[12] L. Gueguen, A. Sergeev, B. Kadlec, R. Liu, and J. Yosinski, "Faster neural networks straight from jpeg," *Advances in Neural Information Processing Systems*, vol. 31, pp. 3933–3944, 2018.

[13] S. Wang, A. Group, H. Lu, and Z. Deng, "Fast object detection in compressed video," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 7103–7112.

[14] S. Jaballah and M. Larabi, "Fast object detection in H264/AVC and HEVC compressed domains for video surveillance," in *2019 8th European Workshop on Visual Information Processing (EUVIP)*, 2019, pp. 123–128.

[15] M. Alizadeh and M. Sharifkhani, "Compressed domain moving object detection based on CRF," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 3, pp. 674–684, 2020.

[16] S. H. Khatoonabadi and I. V. Bajic, "Video object tracking in the compressed domain using spatio-temporal markov random fields," *IEEE Transactions on Image Processing*, vol. 22, no. 1, pp. 300–313, 2013.

[17] S. R. Alvar and I. V. Bajić, "MV-YOLO: Motion vector-aided tracking by semantic object detection," in *2018 IEEE 20th International Workshop on Multimedia Signal Processing (MMSP)*, 2018, pp. 1–5.

[18] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, and T. Darrell, "BDD100K: A diverse driving dataset for heterogeneous multitask learning," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 2633–2642.

[19] "Darknet," https://github.com/AlexeyAB/darknet.

[20] "HEVC reference software," https://HEVC.hhi.fraunhofer.de.