EURASIP Journal on Image and Video Processing

RESEARCH **Open Access**

Deep learning-guided video compression for machine vision tasks

Aro Kim[1], Seung-taek Woo[1], Minho Park[1], Dong-hwi Kim[1], Hanshin Lim[2], Soon-heung Jung[2], Sangwoon Kwak[2] and Sang-hyo Park[1*]

*Correspondence:
s.park@knu.ac.kr

[1] School of Computer Science and Engineering, Kyungpook National University, Daegu, South Korea
[2] Media Research Division, Electronics and Telecommunications Research Institute, Daejeon, South Korea

Abstract
In the video compression industry, video compression tailored to machine vision tasks has recently emerged as a critical area of focus. Given the unique characteristics of machine vision, the current practice of directly employing conventional codecs reveals inefficiency, which requires compressing unnecessary regions. In this paper, we propose a framework that more aptly encodes video regions distinguished by machine vision to enhance coding efficiency. For that, the proposed framework consists of deep learning-based adaptive switch networks that guide the efficient coding tool for video encoding. Through the experiments, it is demonstrated that the proposed framework has superiority over the latest standardization project, video coding for machine benchmark, which achieves a Bjontegaard delta (BD)-rate gain of 5.91% on average and reaches up to a 19.51% BD-rate gain.

Keywords: Video compression, Video coding for machines, Deep learning

1 Introduction

As the volume of video data requiring processing, transmitting, and storing continues to rise, efficient video compression is indispensable. The influence of standardization in this realm is profound. The versatile video coding (VVC) standard [1], representing the forefront of compression technology, stands out due to its remarkable efficiency. Designed to address the escalating demands of video compression, VVC demonstrates substantial bitrate reductions—achieving up to 40% over the high-efficiency video coding (HEVC) standard [2].

The VVC standard represents the current state-of-the-art (SOTA) in video compression models, achieving substantially better coding efficiency than its predecessor in view of compressing general video data. However, directly applying such VVC that is optimized for human vision is not always suitable for video data for machine vision tasks. Recognizing this gap, standardization groups, particularly the moving picture experts group (MPEG), have launched the video coding for machines (VCM) project [3]. This initiative proposes techniques that enhance compression efficiency specifically for machine applications, employing strategies that maintain a higher quality of pixels within regions of interest (ROIs) identified by machine detection than in other regions, using VVC as the underlying codec. By adopting the VCM framework, compression efficiency has improved by 12.73% on the SFU-HW-Objects-v1 [4] and 39.73% on the Tencent Video dataset (TVD) [5] for random access (RA), compared to using VVC alone, demonstrating the effectiveness of tailoring compression techniques to the specific needs of machine vision tasks [6].

MPEG experts and academic researchers have made significant strides in exploring methods to enhance video compression performance for machine vision tasks, primarily through preprocessing steps that consider ROIs identified by machine vision, and temporal and spatial resampling [7–13]. While much of this work has adopted the VVC test model (VTM) as the encoder that uses standard VTM encoding options, it is essential to note that VVC is optimized for human vision tasks, which means its default settings may not be consistently ideal for machine vision applications. In the MPEG VCM, there is a specific approach that converts areas outside of the machine's ROIs to grayscale to increase the coding efficiency of VVC [14]. This approach highlights a gap in research concerning the inefficiency of directly applying standard video compression techniques to machine vision contexts. Given that areas outside the ROI are uniformly processed into a single color, creating a clear boundary between these areas and the main object. Adhering strictly to the traditional use of individual encoding options, for frames comprising a machine-determined ROI and its distinctly colored surroundings, might be inefficient. In addition, the unique characteristics of each input sequence are crucial for machine vision tasks, requiring adaptive encoding strategies. Uniformly applying the same encoding options across all sequences could be problematic, highlighting the necessity for more customized video compression approaches in machine-focused applications.

This paper systematically investigates the effect of encoding options for VCM by toggling it on and off. Building upon these elements, we develop a novel framework, called the deep learning-based adaptive switch networks for machines (DASM). DASM is designed to autonomously select more appropriate encoding tools for each specific video sequence with a deep learning-based selector. The implementation of the proposed encoding framework yields a 5.91% gain in terms of Bjontegaard delta rate (BD-rate), on average, and reaches up to a 19.51% gain compared to the VCM reference software 0.6.0 (VCM-RS 0.6.0) that employs VVC test model software version 12 (VTM 12.0).

2 Related works

2.1 Neural network-based VCM

Recently, the field of image compression has made significant advancements with neural-based approaches [15–21], especially in terms of bitrate savings. These approaches have reached or surpassed the performance of traditional codecs for human's perspective. On the contrary, for machine vision applications, a few studies have explored neural network (NN) methods. Le et al. [22] introduced an end-to-end learned system for image compression designed for machine vision, using a convolutional NN (CNN) with residual blocks for the encoder and decoder components, demonstrating efficient bitrate savings. In contrast, our work focuses on compressing videos. Image codecs primarily handle the compression of spatial information within individual images, lacking the need or capability to address temporal redundancy. Consequently, when addressing video codecs, applying image codecs directly to video codec tasks presents challenges due to these fundamental differences. Kim et al. [23] proposed a multiscale feature compression method that encodes larger scale features followed by smaller scale features in sequence, using an end-to-end trainable model comprising CNN layers. While this method is primarily focused on enhancing feature quality, it does not focus on compressing and evaluating the entirety of preprocessed frames.

2.2 Traditional codec-based VCM

In the VCM project conducted by MPEG, based on the VTM, encoding is performed by reducing the quality in areas outside the ROI detected through machine vision [3]. The current framework for VCM provided by MPEG, VCM-RS 0.6.0, proceeds as illustrated in Fig. 1. Prior to compression, a preprocessing stage is conducted to enhance compression efficiency, emphasizing objects of interest for machine vision tasks.

The preprocessing stage typically includes temporal resampling, spatial resampling, and ROI-based processing. In temporal resampling, video frames are resampled at uniform intervals, effectively reducing the input video frame rate in the temporal domain [12, 13]. This process involves downsampling the original video sequence, which is restored to its original frame rate during the post-processing stage following decoding. Spatial resampling involves downsampling the video in the spatial domain. In the latest version of the MPEG VCM, spatial resampling is set to bypass by default, not engaging in this process. The ROI-based processing operates by employing an object recognition model, processing areas outside the recognized regions at different quality levels to achieve bitrate savings. Background masking is conducted by leveraging object recognition technology based on Faster R-CNN [39]. This technique removes unnecessary background information relevant to machine vision tasks, enhancing coding efficiency. Following the preprocessing phase, the video is compressed using the VTM.

In addition to the efforts in the MPEG VCM, significant research has also leveraged traditional video codecs, such as HEVC and VVC, focusing on applications in machine vision. Building on HEVC, Suzuki et al. [24] introduced a proposed image pre-transforming method that employs NNs to improve image recognition accuracy and lower bitrate. For machine visual analysis, Huang et al. [7] evaluated the degree of importance of each coding tree unit (CTU). By identifying the overlap between the machine ROI and the CTUs within images in the context of VVC, the method allocates bits accordingly.

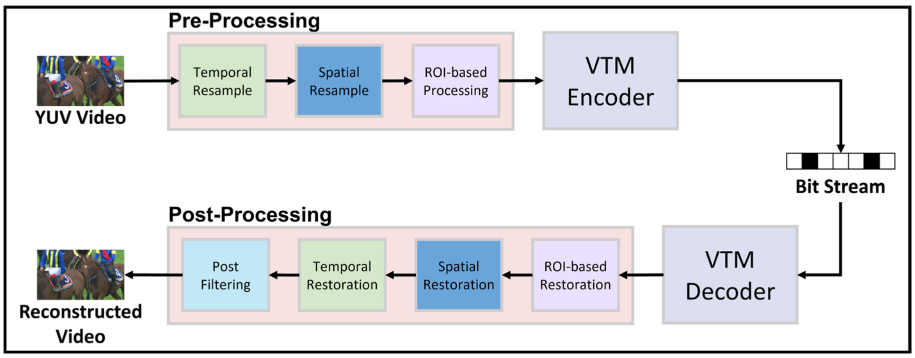

Fig. 1 The overview of VCM framework [3]

Similarly, Fischer et al. [11] built on VVC to determine salient CTUs based on the overlap between detected objects and CTUs. The model adjusts the quantization parameter (QP) values according to whether a CTU is deemed salient, achieving bitrate savings in object detection performance. Likewise, Lee et al. [8] focused on extracting attention regions of machines, allocating bits differently based on these attention regions. The mentioned studies predominantly focus on varying the quality of regions detected by machines, aiming to enhance the efficiency of machine analysis with bitrate savings. This approach is centered on manipulating the output quality for better analytical results rather than exploring the foundational aspects of the codecs themselves.

A few studies have investigated the encoding options of foundational codecs in video compression models designed for machine vision. Fischer et al. [25] explored the influence of three VVC in-loop filters—the deblocking filter, sample adaptive offset filter, and adaptive loop filter—by toggling these options on and off for object detection tasks. Similarly, Lee et al. [9] examined the influence of VVC-based feature coding for machines and its applicability to video captioning, proposing a combination of options for enhanced performance. However, these studies only investigate the effects of encoding options on the specific dataset, reporting the outcomes accordingly. Consequently, they do not provide insight into how these options perform on other input sequences, leaving their efficacy on varied content unexplored.

In the standardization process of the VCM currently at the working draft stage within MPEG [3], several proposals have recently been put forward. Yu et al. [26] suggested a new single layer that applies an NN image codec as an alternative to the conventional VVC intracoding modes for key frame encoding, avoiding the complexities associated with lossless encoding and decoding. Ding et al. [10] proposed truncating the bit depth to enhance the rate–distortion (RD) performance under the VCM common test conditions (CTCs). Lee et al. [27] highlighted the necessity of altering QP selection for encoding video sequences under the CTCs [28], recommending a fixed, constant interval between rate points. Ding et al. [29] introduced a constrained cubic curve-fitting method that achieves a lower fitting error compared to the current approach. However, these submissions are all set against the backdrop of the basic VTM configuration for compression, with no research on the influence of the encoding options on compression for machine tasks.

The VTM provides a range of prediction strategies and coding options. Among these, we investigated the effects of toggling key features, including the deblocking filter, dual I tree, luma mapping and chroma scaling (LMCS), intra-block copy (IBC), and early skip decisions (ESDs), on the encoding process for each input sequence. In VCM, non-ROI regions are converted to grayscale, creating distinct boundaries and sharp contrasts between the objects and the grayscale background. We assumed that since each sequence has varying characteristics and the proportion of the object differs, the adaptive use of these tools based on the frames would impact the compression performance.

First, the deblocking filter is used to smooth the edges of discontinuous blocks, enhancing the quality of the decoded video, particularly when differences in the quantization parameter (QP) across blocks lead to visible discontinuities at block boundaries [30]. In the context of VCM preprocessing, if an object occupies a significant portion of the frame or if there are many objects, it might be more efficient to avoid using the deblocking filter. This can maintain clear boundaries between the object and the grayscale background for better detection while also reducing complexity, thereby improving compression performance. Second, intra-coded I-slices can display distinct spatial characteristics for luma and chroma components. Thus, the dual tree coding structure can be employed to address luma and chroma coding separately, leading to more efficient encoding [31]. This functionality can be managed in the VTM by enabling or disabling the dual I tree option. The dual I tree option could be an efficient choice for camera-captured videos. However, for screen content videos, its effectiveness may vary [32]. In VCM, we assume that the dual I tree is unnecessary when the object is small and the background region is relatively large. This is because introducing a separate tree for chroma can be syntactically inefficient when the simple grayscale occupies a significant portion of the frame. Third, the LMCS is a tool designed to improve encoding performance and subjective image quality [30] by mapping the codewords of the input luma signal to those of the transmitted dynamic range and scaling the chroma signal accordingly. However, in machine vision applications, for frames where grayscale occupies a large portion of the background, encoding luma and chroma separately can introduce inefficiencies and unnecessary syntax. Fourth, the IBC is a technique in which the currently encoded block is predicted from the most similar block within the restored area of the same frame. This approach is anticipated to yield significant BD-rate improvements, particularly in images characterized by repetitive patterns, such as screen video content [33]. In VCM, where non-ROI regions are grayscale and have repetitive patterns, IBC can effectively compress these areas without losing critical information, thereby improving the overall encoding efficiency. Lastly, the ESD is a method that improves performance in terms of speed by determining the inter-frame prediction mode at an early stage. Traditionally, the inter-frame prediction mode was determined by calculating the RD cost for all modes to determine the optimal mode. Each RD cost calculation involves high complexity, leading to the possibility of improving efficiency by determining the mode at an early stage without performing all calculations. In the case of the ESD, after searching for the optimal $2N \times 2N$ mode, if the differential motion vector and coded block flag are zero in the selected optimal mode, no further search is conducted, and it is determined early on to be a skip mode [34]. In the context of VCM, especially for the frames where preprocessing introduced distinct regions, ESD can help quickly identify and skip non-critical regions, thus focusing resources on the ROI. When there is a large grayscale region that is simple, it is likely that the largest block will be selected, making it unnecessary to partition the tree into smaller blocks. Therefore, making early skip decisions can be considered more efficient. This selective processing aligns with the goals of machine vision, where attention needs to be directed primarily at areas of interest.

In Table 1, each encoding option is represented by i. In the VTM 12.0, the default configuration activates the LMCS, deblocking filter, and dual I tree, while disabling the IBC and ESD, as listed in Table 1. The recent VCM-RS 0.6.0 adheres to the default settings of the VTM 12.0 without modifications to specific filters or options. These settings, though primarily tailored to human visual perception, may not align with the objectives of enhancing machine vision performance. Moreover, for every frame processed by the encoder, alternative encoding options might better optimize the balance between the detection accuracy and bitrate efficiency for VCM tasks. There is scant research on the

Table 1 Encoding options represented by *i* and their respective Anchor options

Tool name	Deblocking filter	LMCS	Dual I tree	IBC	ESD
i	0	1	2	3	4
Anchor options	1	1	1	0	0

For each encoding option, 0 in the anchor denotes off, and 1 denotes on

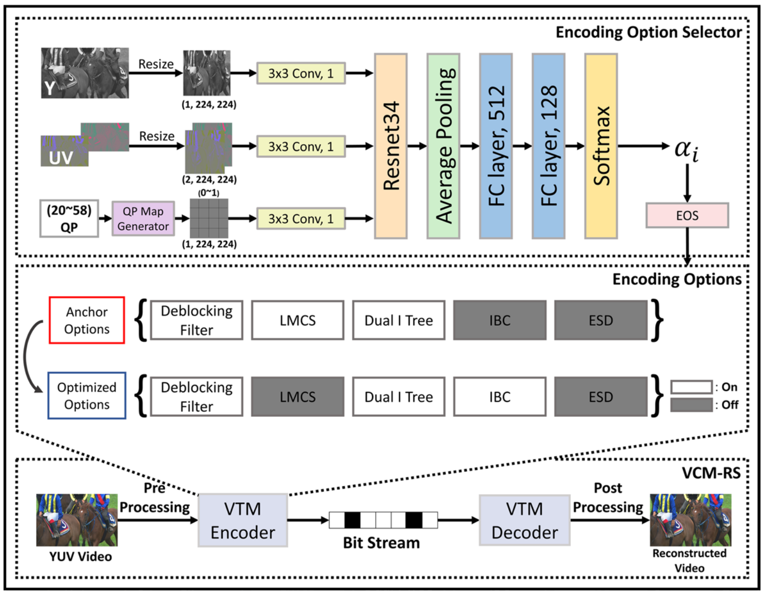

Fig. 2 Detailed architecture of the DASM framework

specific effects of each option within the VCM context. This research distinctively proposes a novel framework, DASM, aimed at determining the most suitable combination of encoding options for each input video. We tailored the encoding process to individual video characteristics, achieving bitrate savings, and advancing encoding technology for machine vision applications.

3 Proposed method

This section details the development process of the DASM framework. This process involves a thorough examination of the encoding options of the VCM-RS. Based on this analysis, we constructed selectors for encoding options. Finally, the approach includes video compression for machines with the optimized options selected. Figure 2 illustrates an overview of the proposed method. The details regarding DASM are described in the following sections.

3.1 Coding options test

The encoding tools in VCM-RS, based on the VTM, include the deblocking filters and LMCS for in-loop filters, dual I tree, and IBC for screen content coding tools, along with adjustments for the ESD. As noted in Section 2.2, VCM-RS follows the default settings of the VTM encoding options for the use of each option. However, the experimental analysis revealed that deviating from the default options of the VTM can result in better compression performance in terms of BD-rate, BD mean average precision (BD-mAP), and BD multiple object tracking accuracy (BD-MOTA). To demonstrate this, we compressed the video datasets from the CTC [28] provided by the MPEG VCM, including the SFU-HW-Objects-v1 [4], TVD [5], and Multiple Object Tracking datasets (i.e., MOT15 [35], MOT17 [36], MOT20 [37]), using the VCM-RS. The VCM-RS 0.6.0 employed for encoding that uses the VTM 12.0, serves as the anchor for the results comparison. We employed TVD, MOT15, MOT17, and MOT20 as the training datasets, and the SFU-HW-Objects-v1 dataset was applied as the testing dataset. The TVD has a resolution of 1920 $\times$ 1080. The resolution of MOT15 is 640 $\times$ 480, and the resolution of both MOT17 and MOT20 is 1920 $\times$ 1080. Unlike traditional codecs, such as HEVC and VVC, which measure the BD-rate and BD peak signal-to-noise ratio (PSNR) based on the bitrate and PSNR graph, the VCM-RS compares the compression performance using the mAP and MOTA instead of the PSNR to suit machine vision tasks better. For training data, we measured MOTA using the JDE-1088 $\times$ 608 [38] model. Based on this, we calculated the BD-rate and BD-MOTA. The Faster R-CNN X101-FPN [39] was employed to measure the mAP of the test data.

When encoding the training and testing sets, we labeled the data based on whether changing the default options for the deblocking filter, LMCS, dual I tree, IBC, and ESD led to improved or degraded results for the BD-rate, BD-mAP, or BD-MOTA compared to the anchor. For example, Table 2 shows the results of encoding the test dataset with the opposite options to the anchor. During BD-rate calculation, errors can occur when the mAP value is nonmonotonic relative to the bitrate. To address this, we also presented Pareto mAP, a metric that selects only the points where the mAP increases monotonically relative to the bitrate for each sequence, ensuring that the BD-rate is calculated without errors. In addition, we encoded the test dataset with LMCS, IBC, and ESD options opposite to the anchor simultaneously, which individually showed better performance over the anchor in Table 2. Table 3 presents the results, showing improved compression performance. These experiments demonstrate that following the anchor option may not be optimal for VCM. Thus, we identified sets of option combinations for each video sequence and QP that yielded improved outcomes, detailed in subsection 3.2.1. In sum, we created ground truth labels for each video sequence depending on the QP. The labels were applied as ground truth to train the encoding option selectors of DASM, as detailed in the following section. Following the CTC [28], encoding was performed in an RA environment, ensuring the broad applicability of the framework across various encoding scenarios. The hardware used for this process includes an Intel core i7-10700 CPU at 2.90 GHz and an Nvidia GeForce RTX 3080 (10 GB) GPU.

3.2 Encoding option selector

The DASM framework, as shown in Fig. 2, comprises encoding option selectors, with each selector designed and trained for individual encoding options. Inputs for these selectors are the QP information and frames in YUV format. In the video sequence, each frame is extracted at a rate of one per second to serve as input. The Y (luminance) and UV (chrominance) components of the frames are separated and input individually, allowing for tailored processing that acknowledges the distinct characteristics and

Table 2 BD-rate (%), Pareto mAP (%) and BD-mAP gains of the opposite options to Anchor over the VCM-RS Anchor on SFU-HW-Objects-v1 for object detection

Sequences		BD-rate(%)				
		i				
		0	1	2	3	4
ClassA	Traffic	–	–	–	–	–
ClassB	ParkScene	–	–	–	–	–
	Cactus	–	–	–	–	–
	BasketballDrive	12.52%	− 7.03%	− 1.98%	− 3.26%	− 8.49%
	BQTerrace	–	–	–	–	–
ClassC	BasketballDrill	5.82%	− 5.56%	− 4.41%	− 0.85%	2.21%
	BQMall	6.12%	− 1.70%	4.93%	− 5.08%	3.44%
	PartyScene	27.86%	− 7.35%	− 11.90%	− 7.10%	− 11.62%
	RaceHorsesC	29.70%	1.83%	8.24%	7.40%	4.65%
ClassD	BasketballPass	19.36%	− 5.20%	5.31%	3.48%	9.23%
	BQSquare	− 6.14%	14.21%	− 1.64%	− 14.30%	− 8.12%
	BlowingBubbles	–	–	–	–	–
	RaceHorses	26.55%	− 2.28%	2.26%	1.88%	− 3.56%
	Average	15.22%	− 1.63%	0.10%	− 2.23%	− 1.53%
Sequences		**Pareto mAP(%)**				
		i				
		0 (%)	**1 (%)**	**2 (%)**	**3 (%)**	**4 (%)**
ClassA	Traffic	− 3.43	− 2.20	− 5.88	0.66	− 9.94
ClassB	ParkScene	41.80	− 7.92	− 6.73	− 16.95	− 6.54
	Cactus	50.52	− 5.89	− 3.95	11.23	0.86
	BasketballDrive	12.52	− 7.03	− 1.98	− 3.26	− 8.49
	BQTerrace	5.43	− 3.12	4.98	6.64	2.48
ClassC	BasketballDrill	5.82	− 5.56	− 4.41	− 0.85	2.21
	BQMall	6.12	− 1.70	4.93	− 5.08	3.44
	PartyScene	27.86	− 7.35	− 11.90	− 7.10	− 11.62
	RaceHorsesC	29.70	1.83	8.24	7.40	4.65
ClassD	BasketballPass	19.36	− 5.20	5.31	3.48	9.23
	BQSquare	− 6.14	14.21	− 1.64	− 14.30	− 8.12
	BlowingBubbles	0.90	16.47	3.64	1.97	2.59
	RaceHorses	26.55	− 2.28	2.26	1.88	− 3.56
	Average	16.69	− 1.21	− 0.55	− 1.10	− 1.76
Sequences		**BD-mAP**				
		i				
		0	**1**	**2**	**3**	**4**
ClassA	Traffic	0.78	0.89	0.75	0.28	1.22
ClassB	ParkScene	− 3.80	− 0.53	1.25	0.59	0.31
	Cactus	− 17.56	3.23	4.13	− 3.05	2.72
	BasketballDrive	− 1.72	1.14	0.35	0.49	1.32
	BQTerrace	− 0.65	0.39	− 0.37	− 0.85	− 0.06
ClassC	BasketballDrill	− 0.42	0.36	0.29	0.05	− 0.16
	BQMall	− 0.71	0.13	− 0.35	0.37	− 0.30
	PartyScene	− 5.13	1.07	2.36	2.08	1.84
	RaceHorsesC	− 2.54	− 0.05	− 0.61	− 0.51	− 0.36

Table 2 (continued)

Sequences		BD-mAP				
		i				
		0	1	2	3	4
ClassD	BasketballPass	− 1.12	0.27	− 0.33	− 0.24	− 0.49
	BQSquare	0.34	-0.85	0.14	0.86	0.43
	BlowingBubbles	0.84	− 1.53	0.41	0.73	− 0.21
	RaceHorses	− 2.33	0.28	− 0.31	− 0.17	0.46
	Average	− 2.62	0.37	0.59	0.05	0.52

BD-rate values marked with '–' represent the cases where the mAP was nonmonotonic to the QP, so the calculation of BD-rate using all six points cannot be made

Table 3 BD-rate (%), Pareto mAP (%), and BD-mAP gains of the opposite options to Anchor over the VCM-RS Anchor of LMCS, IBC and ESD on SFU-HW-Objects-v1 for object detection

Sequences		BD-rate	Pareto mAP (%)	BD-mAP
Class A	Traffic	0.46%	0.46	− 0.05
Class B	ParkScene	–	− 17.34	0.25
	Cactus	–	− 4.97	4.67
	BasketballDrive	− 4.58%	− 4.58	0.70
	BQTerrace	4.94	4.94	− 1.00
Class C	BasketballDrill	0.15%	0.15	− 0.01
	BQMall	3.55%	3.55	− 0.31
	PartyScene	− 4.42%	− 4.42	1.08
	RaceHorsesC	1.28%	1.28	− 0.01
Class D	BasketballPass	3.78%	3.78	− 0.24
	BQSquare	− 14.70%	− 14.70	0.87
	BlowingBubbles	–	12.67	− 1.02
	RaceHorses	− 4.32%	− 4.32	0.46
	Average	− 1.39%	− 1.81	0.41

BD-rate values marked with '–' represent the cases where the mAP was nonmonotonic to the QP, so the calculation of BD-rate using all six points cannot be made

significance of each component in the encoding process. The optimal set of encoding options could vary depending on the QP, even for the same video. To learn and distinguish these variations specific to the QP for the same video, we chose to explicitly input the QP values into the classifier. The QP map generator transforms the QP values, which range from 20 to 58, into a tensor with values ranging from 0 to 1. This tensor was resized to a height and width of 224 before inputting it into each selector. This approach enables each selector to make decisions based on the content of the video frames and QP values, optimizing the encoding process.

The Y and UV information and the QP data initially pass through a convolutional (conv) layer designed to output a single channel, resulting in a total of three channels. However, for the LMCS and dual I tree options, which are significantly influenced by the relationship between the Y and UV components, the relationship was emphasized by processing Y through a 32-channel convolutional layer, UV through a 16-channel layer, and QP through a 16-channel layer, resulting in a total of 64 channels. After the

Y, UV, and QP data are processed through these convolutional layers, they are concatenated and input into a classifying network, based on ResNet34 architecture [40]. This ResNet structure is trained from scratch, without using any pre-trained models. Following this, the network employs a global average pooling layer to output 512 nodes, which pass through two fully connected (FC) layers, one with 512 nodes and another with 128 nodes. Finally, a softmax function is applied to output the activation probabilities. The activation probabilities are input through an encoding option switch (EOS) described in the following subsection, which regulates the switching of encoding options.

3.2.1 Training process of encoding option selector

For data generation, we employed TVD, MOT15, MOT17, and MOT20 as training datasets and SFU as test dataset. To label the datasets, we encoded each dataset by independently changing only one coding tool option from its default or anchor state. This process generated results for the six QPs of all datasets by encoding them oppositely to the default anchor state. For each sequence and each QP, we replaced the results with those encoded oppositely to the anchor and examined the changes in BD-Rate and BD-mAP or BD-MOTA. If the results showed improvements over the anchor for all metrics, we labeled the dataset as 1; otherwise, we labeled it as 0. We repeated this labeling process for all five coding tools across the datasets.

For the training procedure, we trained the Encoding Option Selectors using the aforementioned labeled datasets. The input consisted of frames sampled at one frame per second from each video sequence. The output was the prediction of 0 or 1, with the labeled 0 and 1 serving as the ground truth for training, applying a binary cross-entropy loss function. The training was conducted with a batch size of 64 over 300,000 iterations, employing the Adam optimizer. The learning schedule was cosine annealing warm restarts, with the learning rate starting at 0.001 and decreasing to a minimum of 0.000001, according to the scheduler.

3.3 Encoding option switch (EOS)

Within the DASM framework, the EOS determines whether to activate or deactivate encoding options by comparing the probabilities output from softmax, ranging between 0 and 1, with the predefined threshold values:

$$\phi(\alpha_i) = \begin{cases} 1 & \text{if } \alpha_i > \tau_i \\ 0 & \text{if } \alpha_i \leq \tau_i, \end{cases} \tag{1}$$

In Eq. 1, the EOS corresponds to the function $\phi()$, where α_i represents the probability values obtained from the softmax, and τ_i represents the threshold value for each encoding option.

For a video sequence extracted at one frame per second, if the probability exceeds the threshold, as defined in Eq. 1, in more than half of the frames, the EOS triggers a switch to turn on the VTM options. Conversely, if the probability is below the threshold in more than half of the frames of the video sequence, the switch turns off the option. These settings are relayed to the VTM encoder of VCM-RS, dictating the encoding configuration within the VCM-RS framework. This mechanism ensures that encoding options are dynamically adjusted based on the evaluated probabilities, optimizing the encoding process for enhanced performance and efficiency.

While traditional rounding functions typically round these softmax output probabilities based on a 0.5 threshold, the EOS employs specific threshold values for each encoding option to determine on their activation status. This approach tailors the activation process to the unique requirements and performance sensitivities of each option. In the confusion matrix, considering the default options of the VTM as "true", the labels for each video sequence and QP can either be true or false. A true negative occurs when both the label and prediction represent the reverse of the default option. A false positive arises when the label is the reverse of the default option, but the prediction favors the default option. A false negative is identified when the label supports the default option, but the prediction opts for the reverse. A true positive is noted when both the label and prediction align with the default option. A false negative represents a scenario in which the system incorrectly chooses the reverse of the default or anchor option (a considerable error). Thus, we meticulously adjusted the threshold for each classifier from 0.1 to 0.9, calculating the recall value to minimize the false negatives and selecting the most effective threshold accordingly. Table 4 displays the thresholds set as a result. In addition, experiments regarding the threshold in relation to the F1-score were conducted, with detailed outcomes of these analyses available in the ablation study.

4 Experimental results

4.1 Experimental setup

The SFU-HW-Objects-v1 video dataset was employed to test video compression using the DASM framework. This dataset was accepted as a testing dataset by the MPEG VCM group, comprising four classes with 13 sequences. Specifically, Class A includes Traffic, Class B consists of ParkScene, Cactus, BasketballDrive, BQTerrace, Class C comprises BasketballDrill, BQMall, PartyScene, RaceHorses, and Class D includes BasketballPass, BQSquare, BlowingBubbles, RaceHorses. These classes have resolutions of 2560×1600, 1920×1080, 832×480, and 416×240, respectively. Each sequence was compressed at six QP points, as specified in the CTC [28]. The metric mAP@[0.5:0.95], as described in the previous section, was selected to assess the machine vision performance. All experiments were conducted in an RA configuration based on the VTM 12.0, adhering to the VCM CTC [28].

4.2 Compression results on the video dataset

The DASM framework, proposed for video compression with machine vision considerations, was compared using the VCM-RS 0.6.0, based on the current SOTA video compression model VTM. Table 5 and Figs. 3, 4, 5, and 6 present the results for video resolution-specific BD-rate and BD-mAP, alongside the graphical data obtained using

Table 4 Threshold for the encoding option selector

	i				
	0	**1**	**2**	**3**	**4**
Threshold	0.2	0.1	0.1	0.4	0.8

Table 5 BD-rate (%), Pareto mAP (%), and BD-mAP gains of the proposed method over the VCM-RS Anchor on SFU-HW-Objects-v1 for object detection using Faster R-CNN

Sequences		**BD-rate (%)**	**Pareto mAP (%)**	**BD-mAP**
Class A	Traffic	–	0.19	0.01
Class B	ParkScene	–	0.00	0.13
	Cactus	–	− 1.49	1.90
	BasketballDrive	− 0.48%	− 0.48	0.07
	BQTerrace	–	− 0.75	0.34
Class C	BasketballDrill	− 1.66%	− 1.66	0.11
	BQMall	− 0.14%	− 0.14	0.01
	PartyScene	− 8.62%	− 8.62	1.50
	RaceHorsesC	− 4.40%	− 4.40	0.45
Class D	BasketballPass	− 6.56%	− 6.56	0.33
	BQSquare	− 19.51%	− 19.51	1.09
	BlowingBubbles	–	0.00	0.00
	RaceHorses	− 5.93%	− 5.93	0.70
	Average	− 5.91%	− 3.80	0.51
Sequences		BD-rate (%)	Pareto mAP (%)	BD-mAP

BD-rate values marked with '–' represent the cases where the mAP was nonmonotonic to the QP, so the calculation of BD-rate using all six points cannot be made

Table 6 Model parameters and inference time per encoding option selectors in DASM

Encoding option selector	**0, 3, 4**	**1, 2**
First hidden layer channels	3	64
Model parameter	21, 342, 952	21, 378, 961
Inference time (ms)	835.2	854.6

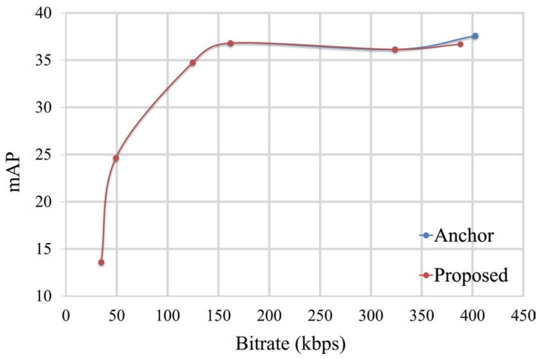

Fig. 3 Rate–distortion curve for the proposed method, compared with the VCM-RS Anchor on SFU-HW-Objects-v1 Class A (Traffic) on object detection tasks

the proposed method. In Table 5, BD-rate values accompanied by asterisks indicate instances when the mAP was nonmonotonic with respect to the QP, preventing the accurate calculation of BD-rate across all six points. Consequently, these values were substituted with figures that did not cause calculation errors and were excluded from the average value computations. Compression via DASM resulted in an average BD-rate

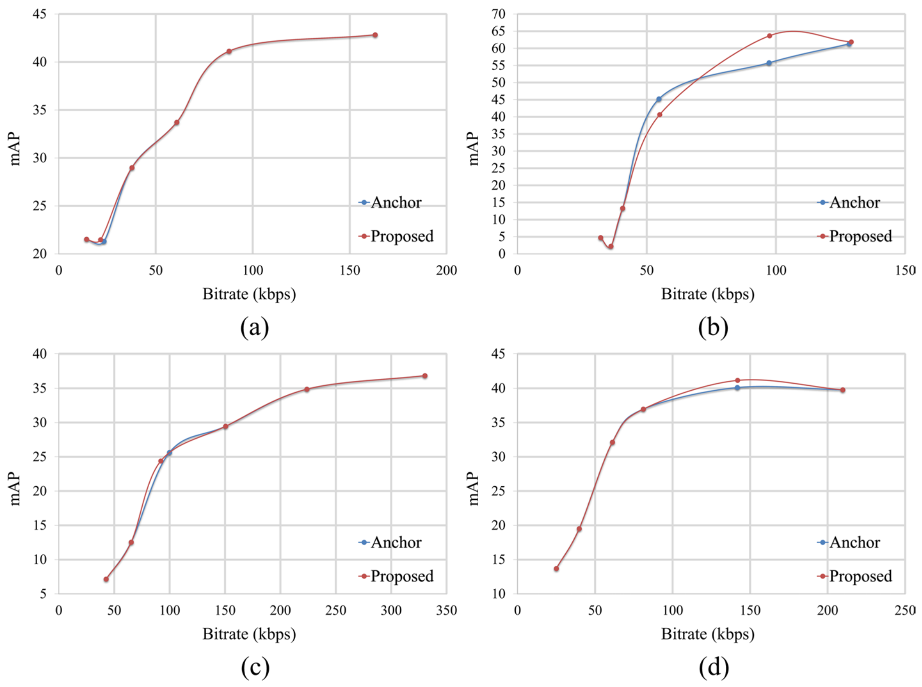

Fig. 4 Rate–distortion curves for the proposed method, compared with the VCM-RS Anchor on SFU-HW-Objects-v1 Class B (ParkScene, Cactus, BasketballDrive, BQTerrace) on object detection tasks

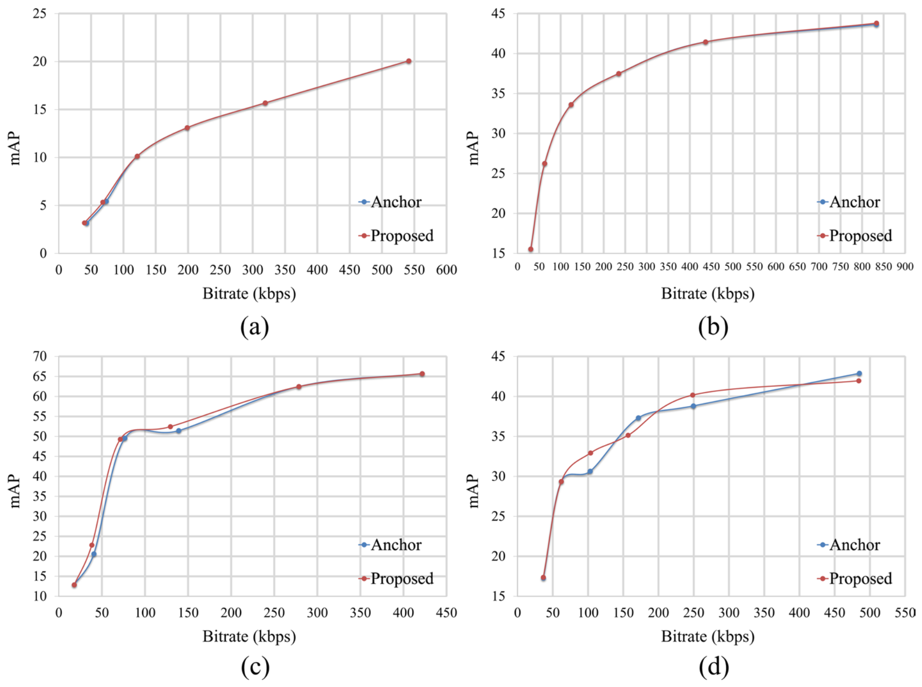

Fig. 5 Rate–distortion curves for the proposed method, compared with the VCM-RS Anchor on SFU-HW-Objects-v1 Class C (BasketballDrill, BQMall, PartyScene, RaceHorses) on object detection tasks

improvement of −5.91%, indicating a high level of compression efficiency. Furthermore, the BD-mAP increased by an average of 0.51, demonstrating improved compression efficiency and enhanced detection performance for machine vision tasks. An improvement

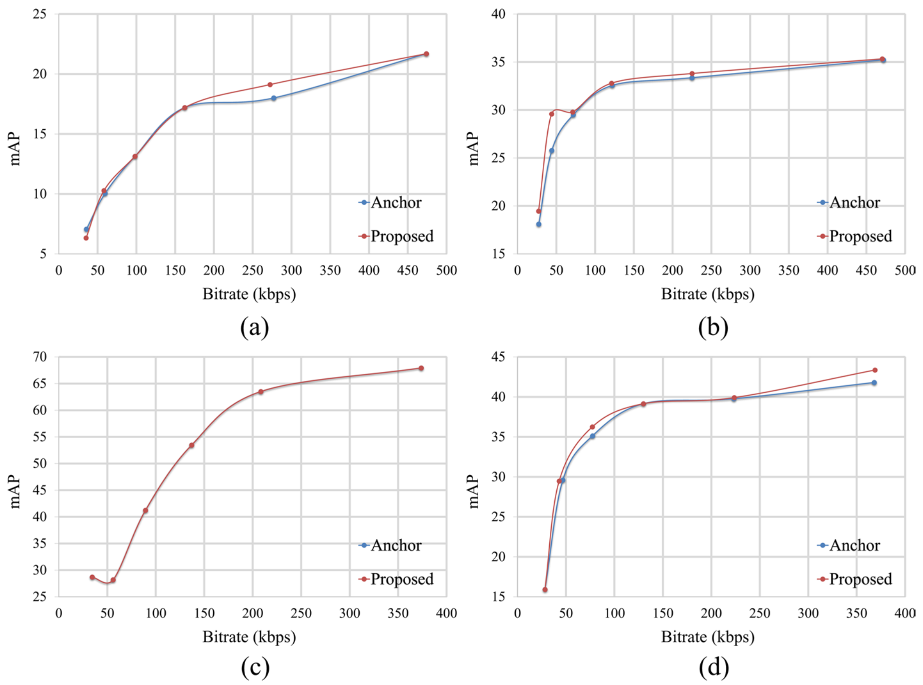

Fig. 6 Rate–distortion curves for the proposed method, compared with the VCM-RS Anchor on SFU-HW-Objects-v1 Class D (BasketballPass, BQSquare, BlowingBubbles, RaceHorses) on object detection tasks

Table 7 Thresholds from the F1-score for encoding option selectors

	i				
	0	**1**	**2**	**3**	**4**
τ	0.2	0.1	0.4	0.7	0.5

in compression efficiency was observed across all resolutions, with notably exceptional efficiency for BQSquare, where the BD-Rate reached −19.51%. In addition, the operation of the encoding option selectors within the DASM framework took an average of only 0.84296 s, exerting a minimal influence on the overall video compression timeline as shown in Table 6.

5 Ablation study

5.1 Compression results from thresholds based on F1 score

This subsection presents the video compression results for DASM using the encoding options selectors trained with threshold values determined by the F1-score, as described in Section 3.3. Table 7 displays the thresholds for the encoding options learned for the selectors. The compression results, in Table 8, indicate an average BD-rate reduction of 2.47. In addition, there was a gain of 0.39 in BD-mAP. In cases where the BD-rate results include asterisks indicate that the mAP is nonmonotonic with respect to the QP, and a non-error-inducing value was substituted when all six points could not be used to determine the BD-rate due to errors. These values were

Table 8 BD-rate (%), Pareto mAP (%), and BD-mAP gains of the proposed method from the F1-score over the VCM-RS Anchor on SFU-HW-Objects-v1 for object detection

Sequences		**BD-rate**	**Pareto mAP (%)**	**BD-mAP**
Class A	Traffic	–	0.19	0.01
Class B	ParkScene	–	10.76	− 0.89
	Cactus	–	− 5.22	3.42
	BasketballDrive	− 1.80%	− 1.80	0.24
	BQTerrace	–	− 1.23	0.17
Class C	BasketballDrill	− 2.59%	− 2.59	0.17
	BQMall	− 0.49%	− 0.49	0.04
	PartyScene	− 6.91%	− 6.91	1.17
	RaceHorsesC	− 6.30%	− 6.30	0.67
Class D	BasketballPass	− 6.56%	− 6.56	0.33
	BQSquare	1.67%	1.67	− 0.06
	BlowingBubbles	–	0.00	0.00
	RaceHorses	3.22%	3.22	− 0.20
	Average	− 2.47%	− 1.17	0.39

BD-rate values marked with – represent cases where the mAP was nonmonotonic to the QP, hindering the accurate calculation of the BD-rate using all six points

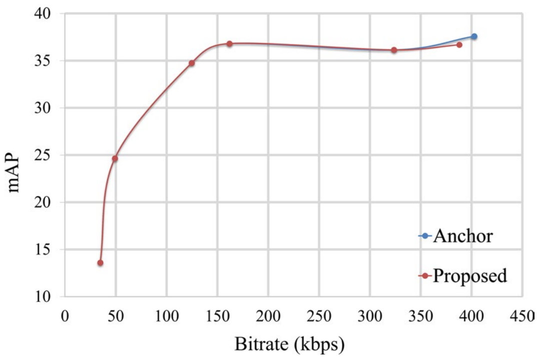

Fig. 7 Rate-distortion curve for the proposed method from the F1-score, compared with the VCM-RS Anchor on SFU-HW-Objects-v1 Class A (Traffic) on object detection tasks

not included in the average calculations. Figures 7, 8, 9, and 10 compare the performance of the proposed method and the anchor across various video resolutions.

5.2 Compression results measured with YOLOv8

In this subsection, we present the compression results obtained using labels derived from thresholds based on recall, evaluated with YOLOv8 [41] instead of the previously used object detection model, Faster R-CNN from CTC. Class D, which contains annotation errors with YOLOv8, was excluded. As shown in Table 9, in Classes A, B, and C, the proposed method demonstrates improvements over the anchor in both BD-Rate and BD-mAP.

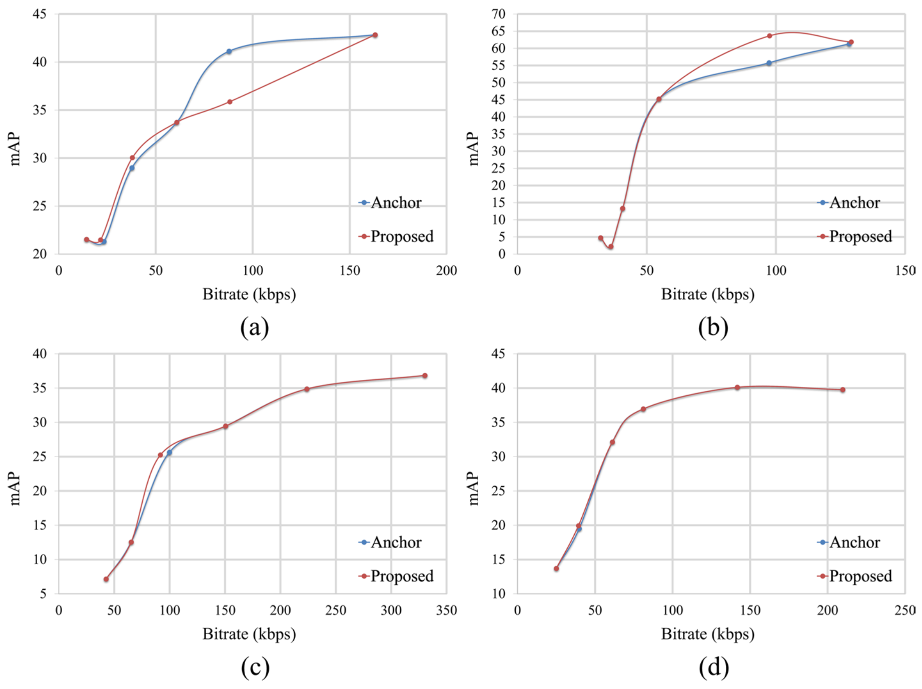

Fig. 8 Rate-distortion curves for the proposed method, compared with the VCM-RS Anchor on SFU-HW-Objects-v1 Class B (ParkScene, Cactus, BasketballDrive, BQTerrace) on object detection tasks

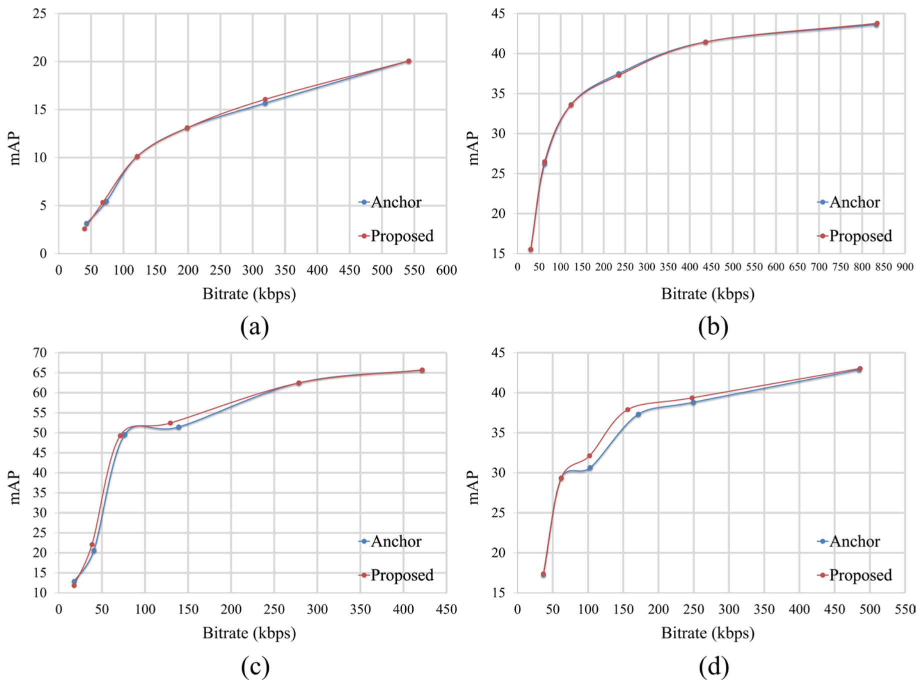

Fig. 9 Rate-distortion curves for the proposed method, compared with the VCM-RS Anchor on SFU-HW-Objects-v1 Class C (BasketballDrill, BQMall, PartyScene, RaceHorses) on object detection tasks

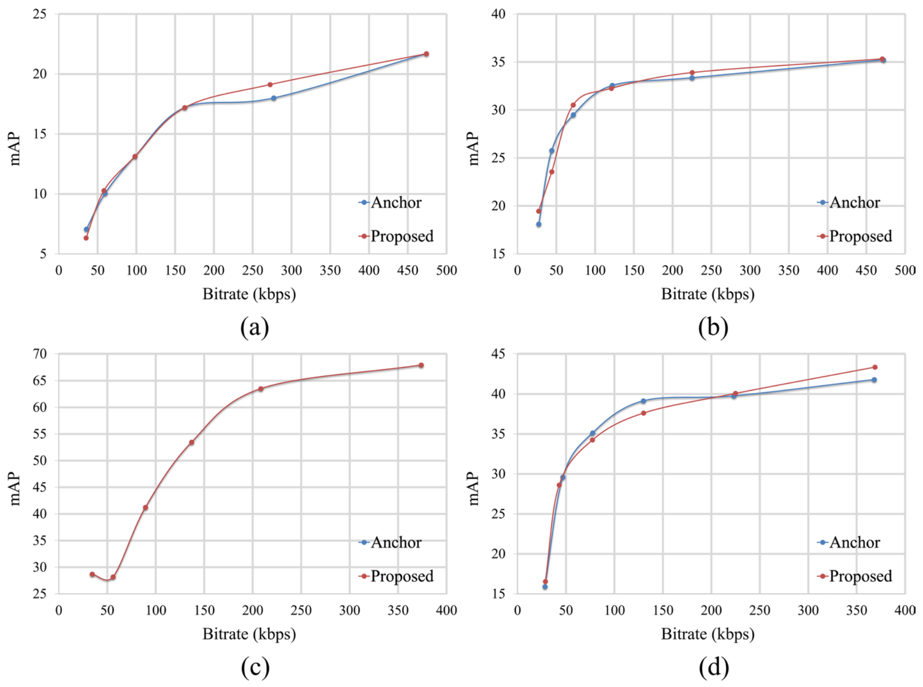

Fig. 10 Rate-distortion curves for the proposed method, compared with the VCM-RS Anchor on SFU-HW-Objects-v1 Class D (BasketballPass, BQSquare, BlowingBubbles, RaceHorses) on object detection tasks

Table 9 BD-rate (%), Pareto mAP (%), and BD-mAP gains of the proposed method over the VCM-RS Anchor on SFU-HW-Objects-v1 for object detection using YOLOv8

Sequences		BD-rate	Pareto mAP (%)	BD-mAP
Class A	Traffic	− 0.18%	− 0.18	0.01
Class B	ParkScene	1.11%	1.11	− 0.23
	Cactus	–	0.00	0.50
	BasketballDrive	− 0.83%	− 0.83	0.24
	BQTerrace	–	0.04	− 0.01
Class C	BasketballDrill	− 2.37%	− 2.37	0.30
	BQMall	0.08%	0.08	− 0.01
	PartyScene	–	− 5.05	0.53
	RaceHorsesC	–	4.42	1.54
	Average	− 0.44%	− 0.31	0.13

BD-rate values marked with '–' represent the cases where the mAP was nonmonotonic to the QP, so the calculation of BD-rate using all six points cannot be made

6 Conclusion

This paper tackles the limitation of VVC coding efficiency, particularly core encoding tools in VVC per video sequence for machine vision tasks, where the video for the machine has very different characteristics from natural video. For that, we propose a novel framework, DASM, for better video compression tailored to machine vision tasks. The proposed approach demonstrates a significant BD-rate reduction of up to 19.51% on the SFU-HW-Objects-v1 dataset. Importantly, this work represents the first endeavor in the research that configures encoding tools to be adaptive to video sequences based on their suitability for machine vision tasks. Thorough experiments show that DASM on top of VCM could enhance compression efficiency across various machine-oriented video sequences without appreciably extending the encoding time.

Abbreviations

BD	Bjontegaard delta
CNN	Convolutional neural network
CTC	Common test conditions
CTU	Coding tree unit
DASM	Deep learning-based adaptive switch networks for machines
EOS	Encoding options switch
ESD	Early skip decisions
HEVC	High-efficiency video coding
IBC	Intra-block copy
LMCS	Luma mapping and chroma scaling
mAP	Mean average precision
MOTA	Multiple object tracking accuracy
MPEG	Moving picture experts group
QP	Quantization parameter
RA	Random access
R-CNN	Regions with convolutional neural network
RD	Rate–distortion
ROI	Region of interests
TVD	Tencent video dataset
VCM	Video coding for machines
VCM-RS	VCM reference software
VTM	Versatile video coding test model
VVC	Versatile video coding

Acknowledgements
The authors would like to thank the experts of ISO/IEC MPEG video for their contributions to VCM.

Author contributions
Conceptualization, S.P.; methodology, S.P. and A.K.; software, S.W.; validation, D.K., M.P., H.L., S.J., and S.K.; investigation, M.P., S.W., A.K., and D.K.; funding acquisition, H.L., S.J., S.K., and S.P.; writing—original draft, A.K., M.P. and S.W.; writing-review and editing, D.K., A.K., and S.W.; supervision, S.P. All the authors have read and agreed to the published version of the manuscript.

Funding
This work was supported in part by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (2020-0-00011, Video Coding for Machine) and in part by BK21 FOUR project (AI-driven Convergence Software Education Research Program) funded by the Ministry of Education, School of Computer Science and Engineering, Kyungpook National University, Korea (4120240214871)

Availability of data and materials
TVD is available at the public repository [https://multimedia.tencent.com/resources/tvd, accessed in April 2024] while SFU dataset can be accessed as shown in CTC [28] but restrictions apply to the SFU dataset.

Declarations

Competing interests
The authors declare that they have no competing interests

Received: 30 April 2024 Accepted: 5 September 2024
Published online: 20 September 2024

References

1. B. Bross, Y.-K. Wang, Y. Ye, S. Liu, J. Chen, G.J. Sullivan, J.-R. Ohm, Overview of the versatile video coding (VVC) standard and its applications. IEEE Trans. Circuits Syst. Video Technol. **31**(10), 3736–3764 (2021)
2. G.J. Sullivan, J.R. Ohm, W.J. Han, T. Wiegand, Overview of the high efficiency video coding (HEVC) standard. IEEE Trans. Circuits Syst. Video Technol. **22**(12), 1649–1668 (2012)
3. S. Liu, C. Rosewarne, Working draft 2 of video coding for machines, document ISO/IEC JTC1/SC29/WG 04, N00465, (2024)
4. SFU-HW-Object-v1, ftp://hevc@mpeg.tnt.uni-hannover.de/testsequences/. Accessed July 2024
5. Tencent video dataset, https://multimedia.tencent.com/resources/tvd. Accessed July 2024

6. S. Różek, O. Stankiewicz, S. Maćkowiak, M. Domański, Video coding for machines using object analysis and standard video codecs, in *Proceedings 2023 IEEE International Conference on Visual Communications and Image Processing (VCIP)*, (Jeju, Korea, 2023), pp. 1–5
7. Z. Huang, C. Jia, S. Wang, S. Ma, Visual analysis motivated rate-distortion model for image coding, in *Proceedings 2021 IEEE International Conference on Multimedia and Expo (ICME)*, (2021), pp. 1–6
8. Y. Lee, S. Kim, K. Yoon, H. Lim, S. Kwak, H.-G. Choo, Machine-Attention-based Video Coding for Machines, in *Proceedings 2023 IEEE International Conference on Image Processing (ICIP)*, (Kuala Lumpur, Malalysia, 2023), pp. 2700–2704
9. J.Y. Lee, Y. Choi, T. Van Le, K. Choi, Efficient feature coding based on performance analysis of Versatile Video Coding (VVC) in Video Coding for Machines (VCM). Multimed. Tools Appl. **82**(27), 42803–42816 (2023)
10. D. Ding, Z. Xin, L. Zizheng, L. Shan, [VCM] Bitwise efficiency: truncating bit depth for machine video coding, document ISO/IEC JTC1/SC29/WG 04, m65525 (Hannover, Germany, 2023)
11. K. Fischer, F. Fleckenstein, C. Herglotz, A. Kaup, Saliency-driven versatile video coding for neural object detection, in *Proceedings 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, (Toronto, Canada, 2021), pp. 1505–1509
12. Z. Lui, Y. Zhang, R. Chernyak, H. Zhu, X. Xu, S. Liu, J. Jia, Z. Chen, M. W. Park, [VCM] Response to VCM call for proposals—an EVC based solution, document ISO/IEC JTC1/SC29/WG 04, m61452 (Mainz, Germany, 2022)
13. L. Yu, VCM-CE 4: temporal resampling, document ISO/IEC JTC1/SC29/WG 04, N00383 (Switzerland, Geneva, 2023)
14. S. Kim, M. Jeong, J. Lee, H. Lee, S. Jung, [VCM] Proposed text of RoI based coding, document ISO/IEC JTC1/SC29/WG 04, m64627 (Switzerland, Geneva, 2023)
15. J. Lee, S. Cho, S.-K. Beack, Context-adaptive entropy model for end-to-end optimized image compression, in *Proceedings 2019 International Conference on Learning Representations (ICLR)*, (New Orleans, USA, 2019), pp. 1–20
16. D. Minnen, S. Singh, Channel-wise autoregressive entropy models for learned image compression", in *Proceedings 2020 IEEE International Conference on Image Processing (ICIP)* (Abu Dhabi, United Arab Emirates, 2020), pp. 3339–3343
17. J. Lin, D. Liu, H. Li, F. Wu, M-LVC: multiple frames prediction for learned video compression, in *Proceedings 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (Seattle, USA, 2020), pp. 3546–3554
18. Z. Cheng, H. Sun, M. Takeuchi, J. Katto, Learned image compression with discretized gaussian mixture likelihoods and attention modules, in *Proceedings 2020 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, (Seattle, USA, 2020), pp. 7939–7948
19. D. He, Y. Zheng, B. Sun, Y. Wang, H. Qin, Checkerboard context model for efficient learned image compression, in *Proceedings 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (Nashville, USA, 2021), pp. 14771–14780
20. Y. Qian, M. Lin, X. Sun, Z. Tan, R. Jin, Entroformer: a transformer-based entropy model for learned image compression, in *Proceedings 2022 International Conference on Learning Representations (ICLR)*, (Virtual, 2022), pp. 1–15
21. J. Li, B. Li, Y. Lu, Neural video compression with diverse contexts, in *Proceedings 2023 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, (Vancouver, Canada, 2023), pp. 22616–22626
22. N. Le, H. Zhang, F. Cricri, R. Ghaznavi-Youvalari, E. Rahtu, Image coding for machines: an end-to-end learned approach, in *Proceedings 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, (Toronto, Canada, 2021), pp. 1590–1594
23. Y. Kim, H. Jeong, J. Yu, Y. Kim, J. Lee, S. Y. Jeong, H. Y. Kim, End-to-end learnable multi-scale feature compression for VCM. IEEE Trans. Circuits Syst. Video Technol. (2023), p. 1
24. S. Suzuki, M. Takagi, K. Hayase, T. Onishi, A. Shimizu, Image pre-transformation for recognition-aware image compression, in *Proceedings 2019 IEEE International Conference on Image Processing (ICIP)*, (Taipei, Taiwan, 2019), pp. 2686–2690
25. K. Fischer, C. Herglotz, A. Kaup, On intra video coding and in-loop filtering for neural object detection networks, in *Proceedings 2020 IEEE International Conference on Image Processing (ICIP)*, (Abu Dhabi, United Arab Emirates, 2020), pp. 1147–1151
26. H. Yu, A. Liu, K. Jia, L, Yu, [VCM] CE2-related: single layer framework with NNIC and VVC for video coding for machines, document ISO/IEC JTC1/SC29/WG 04, m64253, (Geneva, Switzerland, 2023)
27. Y. Lee, K. Yoon, H. Lim, S. Kwak, S. Jung, W, Cheong, H. -G. Choo, [VCM] comments on VCM anchor, document ISO/IEC JTC1/SC29/WG 04, m64809, (Hannover, Germany, 2023)
28. S. Liu, H. Zhang, [VCM] Common test conditions for video coding for machines, document ISO/IEC JTC1/SC29/WG 04, m62002, (2023)
29. D. Ding, H. Wang, X. Zhao, X. Pan, Z. Lui, X. Xu, S. Lui, [VCM] A curve fitting approach to transform nonmonotonic test data for BD-rate calculation, document ISO/IEC JTC1/SC29/WG 04, m65531 (Hannover, Germany, 2023)
30. M. Karczewicz et al., VVC in-loop filters. IEEE Trans. Circuits Syst. Video Technol. **31**(10), 3907–3925 (2021)
31. Y.-W. Huang, J. An, H. Huang, X. Li, S.-T. Hsiang, K. Zhang, H. Gao, J. Ma, O. Chubach, Block partitioning structure in the VVC standard. IEEE Trans. Circuits Syst. Video Technol. **31**(10), 3818–3833 (2021)
32. W. Zhu, J. Xu, L. Zhang, Y. Wang, Adaptive dual tree structure for screen content coding, in *Proceedings 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, (Toronto, Canada, 2021), pp. 1550–1554
33. X. Xu et al., Intra block copy in HEVC screen content coding extensions. IEEE J. Emerg. Sel. Top. Circuits Syst. **6**(4), 409–419 (2016)
34. J. Kim, J. Yang, K. Won, B. Jeon, *Early determination of mode decision for HEVC, in Proceedings 2012 Picture Coding Symposium* (Krakow, Poland, 2012), pp. 449–452
35. L. Leal-Taixé, A. Milan, I. Reid, S. Roth, K. Schindler, Motchallenge 2015: towards a benchmark for multi-target tracking, arXiv preprint arXiv:1504.01942, (2015)
36. A. Milan, L. Leal-Taixé, I. Reid, S. Roth, K. Schindler, MOT16: a benchmark for multi-object tracking, arXiv preprint arXiv:1603.00831, (2016)
37. P. Dendorfer, H. Rezatofighi, A. Milan, J. Shi, D. Cremers, I. Reid, S. Roth, K. Schindler, L. Leal-Taixé, Mot20: a benchmark for multi object tracking in crowded scenes, arXiv preprint arXiv:2003.09003, (2020)
38. Z. Wang, L. Zheng, Y. Liu, Y. Li, S. Wang, Towards real-time multi-object tracking, in *Proceedings 2020 European conference on computer vision (ECCV)*, (2020), pp. 107–122

39. S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: towards real-time object object detection with region proposal networks. Adv. Neural Inform. Process. Syst. **28**, 91–99 (2015)
40. K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in *Proceedings 2016 IEEE conference on computer vision and pattern recognition (CVPR)*, (Las Vegas, USA, 2016), pp. 770–778
41. D. Reis, J. Kupec, J. Hong, A. Daoudi, Real-time flying object detection with YOLOv8, arXiv preprint arXiv:2305.09972, (2023)