PIXEL-LEVEL TEXTURE SEGMENTATION BASED AV1 VIDEO COMPRESSION

Di Chen, Qingshuang Chen, Fengqing Zhu

School of Electrical and Computer Engineering, Purdue University, West Lafayette, Indiana, USA

ABSTRACT

Modern video coding standards use hybrid coding techniques to remove spatial and temporal redundancy. However, efficient exploitation of statistical dependencies measured by a mean squared error (MSE) does not always produce the best psychovisual result. In this paper, we propose a pixel-level texture segmentation approach based on visual relevancy to improve the coding efficiency of newly developed AV1 video codec. Our method performs semantic segmentation and combines regions with similar texture in a video frame. These texture regions are then reconstructed using a motion model at the decoder instead of inter-frame prediction. A Convolutional Neural Networks based semantic segmentation combined with post-processing generates pixel-level texture masks that are more accurate compared to block-based texture masks in our previous work. We show that for many standard test sets, the proposed method achieves significant data rate reductions with improved visual quality.

Index Terms— texture segmentation, convolutional neural networks, video compression, AV1

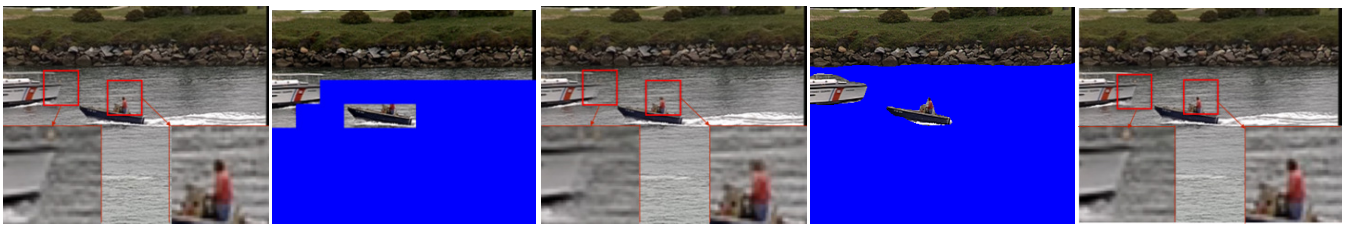
1. INTRODUCTION

Modern video codecs such as HEVC [1] and AV1 [2] use hybrid coding techniques consisting of motion compensation and 2D transform to remove spatial and temporal redundancy. However, efficient exploitation of statistical dependencies measured by a mean squared error (MSE) does not always produce the best psychovisual result. Some regions in the frame, e.g., texture, are perceptually insignificant where an observer does not notice any difference without observing the original video sequence, but are costly to encode. Texture based approaches have been shown to improve the coding efficiency for "perceptually insignificant" regions [3, 4, 5, 6]. We explored similar ideas in our previous work [7], where we do not encode the texture regions, but instead these regions are reconstructed at the decoder based on a motion model. A Convolutioanl Neural Networks (CNN) based texture analyzer was developed to identify the texture regions in a frame and generates block-based texture masks. The displacement of the entire texture region is modeled by a set of motion parameters. At the decoder, instead of performing motion compensation prediction to reconstruct blocks in the texture region, the texture blocks are warped from the reference frames towards the current frame using the motion parameters.

While the proposed approach in [7] can achieve a data rate saving of 1% to 13% compared to the baseline when implemented using AV1 with satisfactory visual quality, the block-based texture masks cannot always accurately represent the texture regions. The block-based texture masks can be seamlessly integrated into AV1 since the common coding units are blocks. However, it can sometimes cause noticeable visual artifacts when an identified texture block consist of small structural region. In addition, the smallest texture block size in [7] was 32×32 in order to avoid detecting small moving objects, but at the same time limits the size of identified texture regions and reduces potential data rate savings. To illustrate this, an example is shown in Figure 1 where part of the bow of the white boat and the person's head are identified as texture region in [7] since the majority of that block is texture. The bow of the white boat and the person's head show flickering artifacts since they have different motion trajectory than the river. There is also some texture regions in the river not identified due to the large block size used in the texture analyzer. The methods proposed in this paper address both of these issues.

Recent advances in deep neural networks have led to a renewed interest in semantic scene segmentation [8, 9, 10]. Large-scale datasest like ImageNet [11], COCO [12] and ADE20K [10] have enabled improved performance for these tasks. For example, the Fully Convolutional Network (FCN) [8] is one of the most commonly used network architectures for semantic scene segmentation. The major issue with FCN [8] is the lack of global contextual information to categories global scene which could lead parsing error. The pyramid scene parsing network(PSPNet) [9] addresses this issue by adding a global pyramid pooling module to extract global information from the image.

In this paper, we incorporate semantic scene segmentation into video compression by generating pixel-level texture segmentation masks to represent "perceptually insignificant" regions in a frame and use motion models to reconstruct the texture regions at the decoder to improve the coding efficiency. We introduce a modified texture mode in AV1 and show that the proposed method can achieve significant data rate reductions with improved visual quality.



(a) (b) (c) (d) (e)

Fig. 1. (a) is the reconstructed frame by AV1 original codec. (b) is the block-based mask. (c) is the reconstructed frame using block-based mask. (d) is the pixel-level mask. (e) is the reconstructed frame using pixel-level mask.

2. PIXEL-LEVEL TEXTURE SEGMENTATION

Our previous work [7] using block-based texture analyzer has shown data rate savings in texture regions. However, block-based texture mask cannot accurately represents texture regions and may cause coding artifacts. Therefore, a pixel-level texture mask generation is described in this section to obtain more accurate texture masks. First, we use a semantic scene segmentation method described in Section 2.1 to generate masks for different semantic classes. Then in Section 2.2, we describe how several semantic classes with similar texture are grouped into four texture classes to produce a single pixel-level segmentation mask for each texture class.

2.1. Semantic Scene Segmentation

Stuff and objects are two high-level categories often used in semantic scene understanding [9, 13] . Stuff refers to background areas such as sky, grass, and they usually contain large texture areas, and objects are more likely to appear in the foreground.

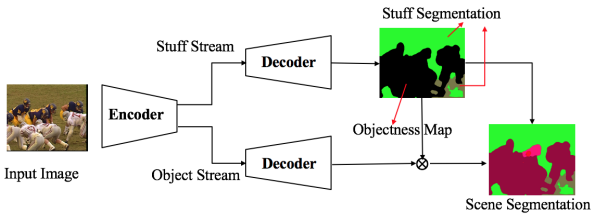


Fig. 2. Two steam cascade framework.

In this paper, we use a two-stream cascade network [13] to generate semantic scene segmentation shown in Figure 2. The stuff stream generates stuff segmentation and objectness map, while the object stream generates object segmentation using the objectness map from the stuff stream. The final segmentation combines the results from stuff stream and object stream. The ResNet50 [14] with dilated convolutions [15, 16] strategy is used as encoder to extract feature map. For the decoder, we use pyramid pooling module from PSPNet [9] followed by bilinear upsampling. The pyramid pooling module uses four different sizes of CNN receptive field to represent global contextual information contained in four pyramid scales. The pyramid pooling module reduces the scene parsing errors by considering global contextual relationship in a scene. Cross-entropy loss is used at the end of each stream. The total training loss is the stuff steam loss plus the object stream loss as expressed in Equation 1, where for a givel pixel x, N is the number of stuff classes and M is the number of object classes, $p_{x,i}$ is the predicted probability of pixel x for class i and $y_{x,i}$ is the binary indicator (0 or 1) for that class.

$$\mathcal{L}_{total} = -\sum_{i=1}^{N} y_{x,i} \log(p_{x,i}) - \sum_{j=1}^{M} y_{x,j} \log(p_{x,j}) \qquad (1)$$

The pre-trained model was obtained from a scene parsing benchmark, MIT ScenParse150 [13], to generate semantic segmentation. The model was trained on a subset of a densely annotated dataset, ADE20K [10], with top 150 categories ranked by their pixel ratios in which 35 of them are stuff classes, 115 are object classes. The pixel accuracy of this model is 80.23% as reported on the benchmark [13].

2.2. Texture Mask Grouping

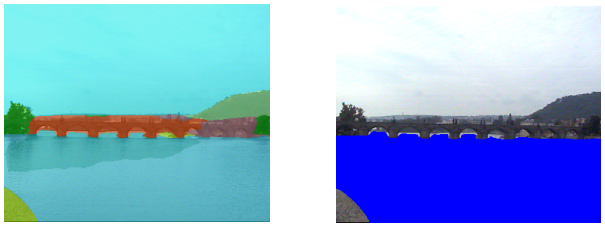
We define four perceptually insignificant texture classes that are commonly observed in nature scenes. The four texture classes are based on groupings of different semantic classes defined in ADE20K dataset [10] that have similar textures. Texture class 1 includes earth and grass semantic classes; texture class 2 includes water, sea and river semantic classes; texture class 3 includes mountain and hill semantic classes; texture class 4 includes tree semantic class. A single pixel-level mask for each texture class is generated by combining semantic segmentation within each group. Figure 3 shows an example of pixel-level texture segmentation mask for texture class 2, which combines semantic segmentations of water and river.

3. MODIFIED TEXTURE MODE IN AV1

In this section, we describe how we modified the texture mode we introduced to AV1 in [7] to incorporate the pixel-level texture masks described in Section 2.

3.1. Modified Texture Mode Encoder Design

Figure 4 illustrates the design of integrating pixel-level texture masks into AV1 encoding. The encoder fetches the pixel-level texture masks for the current frame and selected reference frames. Based on the texture region in the current frame



(a) Semantic segmentation (b) Texture mask for class 2

Fig. 3. An example of pixel-level texture segmentation for video sequence *bridgefar*. Texture mask for class 2 contains semantic segmentations of water and river in this example.

indicated by the texture masks, a set of texture motion parameter that represents the global motion of the texture region is calculated for each reference frame. We "overlap" the pixel-level texture masks on the current frame and reference frame to check if a block is completely covered by the texture mask. If it is, the block is considered a texture block and uses the texture mode as shown in Figure 5. For blocks coded using the texture mode, pixels within the blocks are reconstructed at the decoder using the estimated texture motion parameters for the entire texture region, thus no motion prediction residuals need to be coded and transmitted. Texture blocks are reconstructed by warping the texture region from the reference frame towards the current frame. At the decoder, the bitstream is decoded the same as AV1 baseline since there is no syntax change to the AV1 bitstream and the reconstruction of the texture blocks are performed outside the codec.

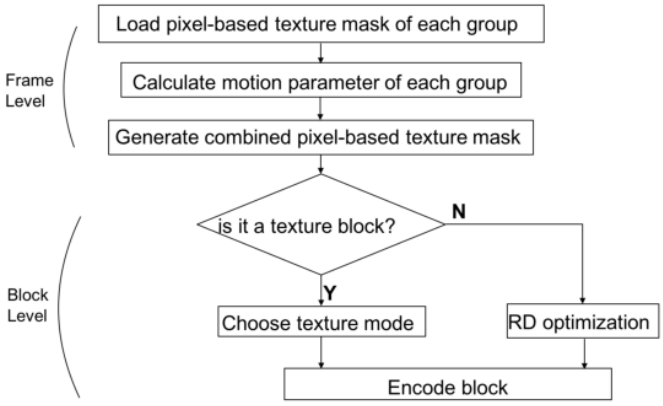


Fig. 4. Design of texture mode encoder

In [7], based on the selection of coding orders and choices of reference frames for texture region reconstruction, we investigated three different implementations and found the best implementation with respect to data rate savings and perceived quality to be *tex-cp*, which we use in this paper. The coding structure of *tex-cp* has fixed group interval of eight frames. It enables texture mode for every other frames. The texture blocks use the previous and the next frames as reference frames and take the average of two warped region as prediction.

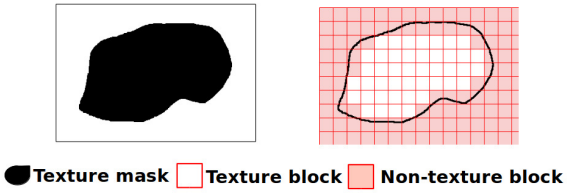


Fig. 5. Texture mode decision

3.2. Motion Model for Texture Regions

In our previous work [7], the block-based texture masks can only differentiate between texture vs. non-texture regions. We select only the largest texture region in a frame to be considered for encoding using the texture mode. We used the global motion tool [17] in the AV1 codec to perform block warping for the texture regions. Using the AV1 codec syntax, one set of motion parameters for each reference frame is sent to the decoder in the compressed frame header.

As described in Section 2, the pixel-level segmentation method can generate up to four pixel-level non-overlapping texture masks for each texture class per frame. In this paper, we allow the texture masks to contain more than one class of texture without changing the codec syntax. The encoder estimates a set of motion parameters for each class of texture. We combine texture masks that have the same set of motion parameters and use one set of motion parameters to represent the largest combined texture masks.

4. EXPERIMENTAL RESULTS

To evaluate the performance of the proposed method using pixel-level texture mask, data rate savings at four quantization levels (QP=16, 24, 32, 40) are calculated for low and high resolution videos from standard video test sequences. We use the original AV1 codec as the baseline for comparison. We also compare the pixel-level texture segmentation results with our previous work [7] which uses block-based single class texture mask with our AV1 texture mode. All three methods use the same golden frame group structure that has fixed golden frame and a group interval of eight frames. Results for the test videos are shown in Table 1. BM in the tables refers to block-based texture segmentation method [7] and the PM refers to the proposed pixel-level texture segmentation method. The data rate saving is calculated as

$$P_{bit} = (R - R_b)/R_b \times 100\% \quad (2)$$

where P_{bit} represents data rate saving, R represents the bitstream size using BM or PM method, R_b represents the bitstream size of the AV1 baseline method. A negative value indicates a reduction in the codec's bitstream data rate compared to the AV1 baselines.

In general, compared to the AV1 baseline, the coding performance of the both BM and PM shows larger data rate savings with low QP. However, as QP increases, the data rate saving decreases. As shown in the tables, *football*, *waterfall* and

Table 1. AV1 data rate savings comparison between block-level (BM) and pixel-level (PM) texture segmentation. A negative value indicates a reduction in the codec's bitstream data rate compared to the AV1 baselines.

Resolution	Video	Data rate saving(%)							
		QP=16		QP=24		QP=32		QP=40	
		BM [7]	PM [proposed]	BM	PM	BM	PM	BM	PM
low res	*coastguard*	-7.8	-9.14	-6.99	-8.01	-4.7	-5.72	-1.9	-2.13
	flower	-10.55	-13	-8.66	-10.78	-5.96	-4.95	-4.95	-4.95
	football	-0.35	-0.63	0.02	-0.08	0.01	0.01	0.02	0
	waterfall	-4.63	-13.11	-3.96	-7.21	0.33	-1.3	3.74	3.48
	netflix_aerial	-8.59	-9.15	-2.15	-5.59	0.68	-1.05	4.59	4.01
high res	*intotree*	-5.32	-9.71	-4.32	-9.42	-1.99	-8.46	2.83	-4.92
	tractor	-3.32	-5.45	-2.25	-4.40	-1.68	-3.90	0.3	-2.93

netflix_aerial have worse coding performance than the AV1 baseline at high QP. The reason is that at high QP, the high compression ratio results in many zero residual blocks thus there is limited margin for data rate saving using texture based methods. In addition, the texture based method requires a few extra bits for the texture motion parameters, and some extra bits for using two reference frames in compound prediction for all the texture blocks.

Compared to the BM method, the proposed PM method shows larger data rate savings. For the BM method, the fixed size blocks for CNN based texture analyzer need to be large enough to ensure classification accuracy. While for the PM method, there is no such limitation so we use 16×16 as the minimum size for texture blocks instead of 32×32 in the texture mode. Therefore, there are more pixels in a frame that are reconstructed using the texture mode in the PM method leading to larger data rate savings. We did not using smaller texture blocks because further block splitting will require extra bits to send the motion information for these blocks. Table 2 shows the texture region percentage, defined in equation 3 as average percentage of the regions that are reconstructed using texture mode within the frames where texture mode is enabled.

$$P_{tex} = (\sum_{j=1}^{F_{tex}} (\frac{\sum_{i=1}^{N_j} B_{ij}}{W \times H}))/F_{tex} \times 100\% \quad (3)$$

where F_{tex} is the number of frames that enables texture mode, N_j is the number of texture blocks in the j^{th} frame, B_{ij} is the block size of texture block i in frame j, W and H are frame width and height.

In general the texture region percentage of PM method is larger than that of the BM method, thus the increase in data rate saving. The texture region percentage of PM for *flower* is smaller because the texture mask of BM contains sky and flowerbed area as it fails to identify them as two different classes of texture. Although texture mask of PM only contains flowerbed area, the sky area is very homogeneous which has small residual using AV1 baseline. Therefore, we still achieve more data rate saving using PM than BM. The PM method also reduces flickering artifacts in some of the reconstructed video when using the BM method. The pixel-level texture mask can more accurately represent the perceptually insignificant pixels. An example is illustrated in Figure 1 and discussed in Section 1.

Table 2. Texture region percentage

Texture region encoded	BM (%)	PM (%)
coastguard	37	41
flower	58	24
football	10	22
waterfall	61	77
netflix_aerial	37	53
intotree	43	52
tractor	20	23

We have conducted a subjective test in our previous work [7] to evaluate the visual quality of texture based coding method. Results show that most of the times (77%), the viewer can not tell the difference between the reconstructed video by the original codec and the BM method. We observed the quality of the reconstructed video by the PM method and compare it with that of the baseline and the BM method. There is no noticeable artifacts and the flickering artifacts in the BM method due to inaccurate texture masks has been significantly reduced.

5. CONCLUSION

In this paper, we improve the coding efficiency of AV1 codec by introducing a texture segmentation based approach that uses deep neural networks to perform pixel-level segmentation which identifies the texture region in a frame. The texture regions are then reconstructed based on a global motion model instead of using inter-frame prediction. We combined semantic segmentation with a few post-processing steps to generate a pixel-level texture mask that is more accurate than our previously proposed block-based texture method. We show that for many standard test sets, the proposed method achieved significant data rate reductions with improved visual quality.

6. REFERENCES

[1] Gary J. Sullivan, Jens Rainer Ohm, Woo Jin Han, and Thomas Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, 2012.

[2] Y. Chen, D. Murherjee, J. Han, A. Grange, Y. Xu, Z. Liu, S. Parker, C. Chen, H. Su, U. Joshi, C. Chiang, Y. Wang, P. Wilkins, J. Bankoski, L. Trudeau, N. Egge, J. Valin, T. Davies, S. Midtskogen, A. Norkin, and P. de Rivaz, "An overview of core coding tools in the av1 video codec," *2018 Picture Coding Symposium (PCS)*, pp. 41–45, June 2018.

[3] Marc Bosch, Fengqing Zhu, and Edward J Delp, "Segmentation-Based Video Compression Using Texture and Motion Models," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 7, pp. 1366–1377, nov 2011.

[4] J. Balle, A. Stojanovic, and J. Ohm, "Models for static and dynamic texture synthesis in image and video compression," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 7, pp. 1353–1365, Nov 2011.

[5] F. Zhang and D. R. Bull, "A parametric framework for video compression using region-based texture models," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 7, pp. 1378–1392, Nov 2011.

[6] F. Racapé, O. Déforges, M. Babel, and D. Thoreau, "Spatiotemporal texture synthesis and region-based motion compensation for video compression," *Signal Processing: Image Communication*, vol. 28, no. 9, pp. 993–1005, Oct. 2013.

[7] D. Chen, C. Fu, and F. Zhu, "AV1 Video Coding Using Texture Analysis With Convolutional Neural Networks," *ArXiv e-prints*, Apr. 2018.

[8] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," *Proceedings of the IEEE Conference On Computer Vision and Pattern Recognition*, pp. 3431–3440, June 2015, Boston, MA.

[9] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 2881–2890, July 2017, Honolulu, HI.

[10] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Semantic understanding of scenes through the ade20k dataset," *arXiv preprint arXiv:1608.05442*, 2016.

[11] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al., "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[12] T. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C L. Zitnick, "Microsoft coco: Common objects in context," *Proceedings of the IEEE European Conference on Computer Vision*, pp. 740–755, September 2014, Zurich.

[13] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Scene parsing through ade20k dataset," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, no. 2, pp. 4, July 2017, Honolulu, HI.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, June 2016, Las Vegas, NV.

[15] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected crfs," *arXiv preprint arXiv:1412.7062*, 2014.

[16] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *arXiv preprint arXiv:1511.07122*, 2015.

[17] S. Parker, Y. Chen, D. Barker, P. de Rivaz, and D. Mukherjee, "Global and locally adaptive warped motion compensation in video compression," *Proceedings of the IEEE International Conference on Image Processing*, pp. 275–279, September 2017, Beijing, China.