INTERNATIONAL ORGANISATION FOR STANDARDISATION
ORGANISATION INTERNATIONALE DE NORMALISATION
ISO/IEC JTC 1/SC 29/WG 4
MPEG VIDEO CODING

ISO/IEC JTC 1/SC 29/WG 4 m62235
January 2023, Online

Title: **[VCM] CE1.6 - RoI-based simplification and retargeting for Video Coding for Machines**

Source:
Marek Domański, Olgierd Stankiewicz, Sławomir Maćkowiak, Tomasz Grajek, Maciej Wawrzyniak, Jakub Stankowski, Sławomir Różek, Dominik Cywiński, Jakub Szekiełda, Jakub Siejak

Poznań University of Technology, Poznań, Poland

Abstract

1 Introduction

This document describes RoI-based simplification and retargeting method implemented in the VCM-RS software [1]. The method originally has been implemented in Poznan University of Technology responses (A,B and C) [2,3,4] to "Call for Proposals for Video Coding for Machines" [CFP]. The document describes also the syntax of the respective bitstream. In the Annex, the source code is provided.

2 General description

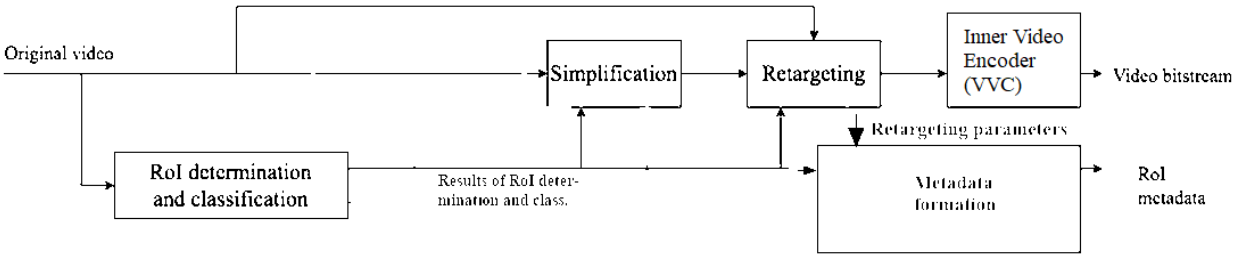

Fig. 1. The structure of the encoder

Description of the building blocks in a VCM encoder:

a) ***RoI determination and classification***: In this block the original image/video is analyzed and potential regions of interest (RoI) are determined in the areas where interesting objects are located. This is done using a special deep neural network, specifically: detectron network. Then the RoIs are classified as corresponding to important objects, as corresponding to small objects and maybe into other classes.

b) ***Simplification***: The simplification is aimed at production of the content that may be represented with reduced number of bits after compression without any significant deterioration of the efficiency of the machine vision tasks executed on decoded (decompressed) image/video. In principle the simplification reduces details in the background whereas the objects and their neighborhoods remain untouched or only slightly simplified. In particular, the results of RoI determination and classification are used to identifying objects, the bands around objects and the background. Here, we do consider the objects already replaced by simple content by the inpainting. The background is simplified, and possibly some bands around selected objects or even some objects (usually the objects of big size – relatively in an image or in a video frame). The goal of simplification is to produce such image/video that can be easily represented by a very small number of bits (in compressed representation) whereas the object still can be well recognized (detected, tracked etc.). This may be optimized in context of particular class of object within given RoI.

c) ***Retargeting***: The number of lines and columns of the input image or the video frame may be reduced according to the content identified in the process of RoI determination and classification. The lines and columns of samples are mostly removed when their do not comprise samples of objects. The lines and rows that comprise object samples may be reduced according to analysis of the objects. This reduction correspond to downsampling the object that must not deteriorate the ability to recognize the objects, therefore it is possible if the object is oversampled with respect of its texture and other features. The sample (line / column) removal must be preceded by the respective local lowpass (anti-aliasing) filtering. In classical retergatting methods energy function has to be formulated which controls which rows and columns are primarily removed. Such would require signalization of the energy function (image) to the decoder. In our proposal a simple yet efficient retargeting technique is used which omits implicit energy image formulation, and instead rectangle-shaped RoI are used. Therefore, only coordinates of RoIs are transmitted.

3 Implementation details

The tool has been implemented as ROI component in VCM-RS [1] software.

- encoder_poznanroi.py – main encoder component file
- decoder_poznanroi.py – main decoder component file
- poznanroi_config_codec.py – hard-coded parameters of the codec e.g. bluring parameters.
- poznanroi_config_classes.py – classes of objects used in the encoder
- poznanroi_detectron.py – interface to detectron network
- poznanroi_blur.py – implementation of simplification (blurring) algorithm
- poznanroi_retarget.py – implementation of retargeting algorithm
- poznanroi_utils.py – utilities related to general processing.
- poznanroi_rois.py - utilities related to RoIs
- poznanroi_objs.py - utilities related to objects
- poznanroi_bitenc.py – bitstream encoding utilities
- poznanroi_bitdec.py – bitstream decoding utilities

The current implementation of the tools has additional requirements:

- Detectron framework
- Bitstring python library – bitstream encoding of RoIs

It can be noted that the current implementation does NOT require GPU for operation (CPU only). It however employs GPU if available to speed-up processing. We have notices negligible differences in encoder performance when CPU is used vs GPU.

4 Syntax

Table 1. VCM RoI information syntax.

Syntax element	C	Descriptor
VCM_info(payloadSize) {		
bits_size_x	5	u
bits_size_y	5	u
output_image_size_x_minus_1	bits_size_x	u
output_image_size_y_minus_1	bits_size_y	u
If (codec_option_use_roi_retargetting) {		
roi_retargetting_info( )		
}		

bits_size_x - number of bits on which output_image_size_x_minus_1 syntax element is signalled.

bits_size_y - number of bits on which output_image_size_y_minus_1 syntax element is signalled.

output_image_size_x_minus_1 - size of the original image to be outputted from the VCM decoder, minus 1.

output_image_size_y_minus_1 - size of the original image to be outputted from the VCM decoder, minus 1.

codec_option_use_roi_retargetting - flag active in all Poznan University of Technology responses to CfP: A, B and C

Table 2. RoI retargeting information syntax.

Syntax element	C	Descriptor
roi_retargetting_info() {		
hierarchy_power_base	8	u
if (hierarchy_power_base ! = 0) {		
bits_scale	3	u
bits_pos_x	5	u
bits_pos_y	5	u
bits_size_x	5	u
bits_size_y	5	u
bg_scale	bits_scale	u
bg_size_x_minus_1	bits_pos_x	u
bg_size_y_minus_1	bits_pos_y	u
bits_num_rois	5	5
num_rois	bits_num_rois	u
current_scale = 0		
for (i = 0; i < num_rois; i++) {		
if (current_scale!=0) {		
update_scale	1	u
if (update_scale):		
current_scale	bits_scale	u
}		
roi_scale[i] = current_scale		
roi_pos_x[i]	bits_pos_x	u
roi_pos_y[i]	bits_pos_y	u

roi_size_x_minus_1[i]	bits_size_x	u
roi_size_y_minus_1[i]	bits_size_y	u
}		
}		

hierarchy_power_base - base of power function basing on which retargetting scaling is calculated

bits_scale - number of bits on which bg_scale and roi_scale[i] syntax elements are signaled

bits_pos_x - number of bits on roi_pos_x[i] syntax element is signaled

bits_pos_y - number of bits on roi_pos_x[i] syntax element is signaled

bits_size_x - number of bits on roi_size_x_minus_1[i] syntax element is signaled

bits_size_y - number of bits on roi_size_y_minus_1[i] syntax element is signaled

bg_scale - scale level of the background (highest hierarchy RoI element)

bg_size_x_minus_1 - size (minus 1) of the background (highest hierarchy RoI element)

bg_size_y_minus_1 - size (minus 1) of the background (highest hierarchy RoI element)

bits_num_rois - number of bits on which num_rois syntax elements is signaled

num_rois - number of RoIs (excluding background)

current_scale - temporary decoding variale, used to update successive scales: roi_scale[i]

roi_scale[i] - scale of RoI index i

roi_pos_x[i] - position of RoI index i

roi_pos_y[i] - position of RoI index i

roi_size_x_minus_1[i] - width (minus 1) of RoI index i

roi_size_y_minus_1[i] - height(minus 1) of RoI index i

5 The experimental results

Table 1. Coding results on OpenImages detection and segmentation evaluation subsets

Machine task	QP	Anchor (VTM 12.0, scale 100%)		Our proposal	
		BPP	mAP	BPP	mAP
Detection	22	0.863	78.929	0.220	76.720
	27	0.509	77.989	0.135	75.664
	32	0.287	77.263	0.081	74.059
	37	0.153	73.963	0.048	69.980
	42	0.078	68.842	0.027	63.069
	47	0.037	58.021	0.015	48.958
Segmentation	22	0.841	80.536	0.212	78.960
	27	0.493	80.197	0.129	77.497
	32	0.277	78.775	0.077	75.159
	37	0.147	75.653	0.045	71.409
	42	0.074	69.917	0.026	63.987
	47	0.036	57.773	0.014	49.748

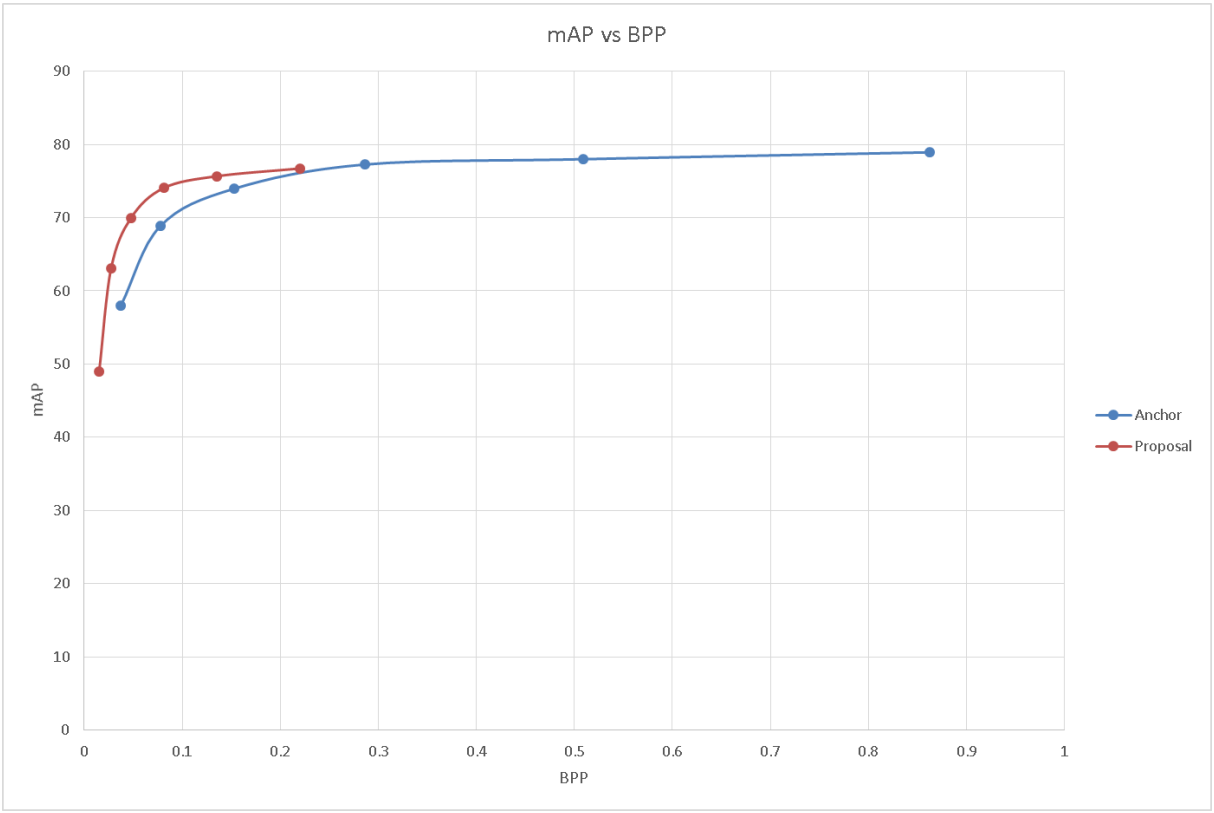

Fig. 2. Detection results on OpenImages dataset (evaluation subset for detection)

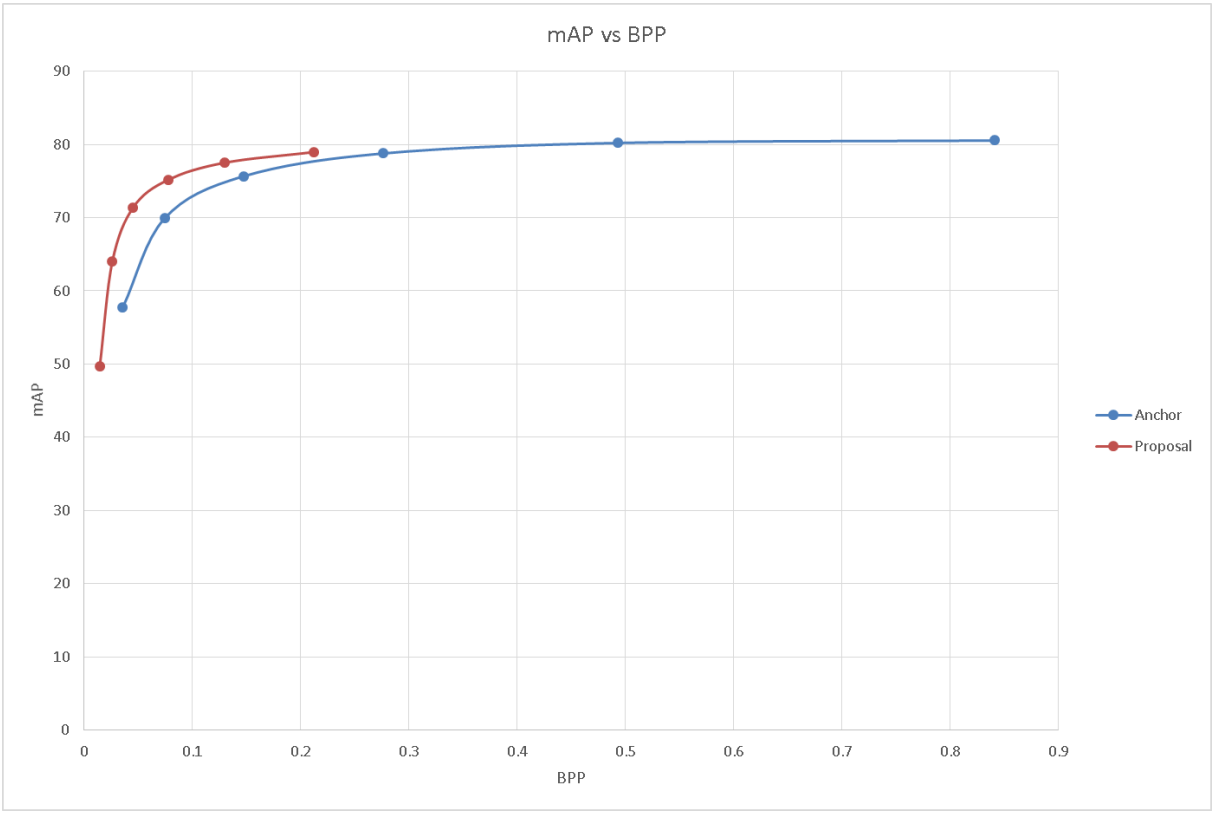

Fig. 3. Segmentation results on OpenImages dataset (evaluation subset for segmentation)

6 References

[1] Honglei Zhang , “Introduction to the VCM reference software (VCM-RS) “,ISO/IEC JTC 1/SC 29/WG 4 m62003, January 2023, Online

[2] Marek Domański, Olgierd Stankiewicz, Sławomir Maćkowiak, Sławomir Różek, Tomasz Grajek, Jakub Szekiełda, Dominik Cywiński, Jakub Siejak, "[VCM] Poznań University of Technology Proposal A in response to CfP on Video Coding for Machines", ISO/IEC JTC 1/SC 29/WG 2 m60727, Online – October 2022.

[3] Marek Domański, Olgierd Stankiewicz, Sławomir Maćkowiak, Sławomir Różek, Tomasz Grajek, Jakub Szekiełda, Dominik Cywiński, Jakub Siejak, "[VCM] Poznań University of Technology Proposal B in response to CfP on Video Coding for Machines", ISO/IEC JTC 1/SC 29/WG 2 m60728, Online – October 2022.

[4] Marek Domański, Olgierd Stankiewicz, Sławomir Maćkowiak, Sławomir Różek, Tomasz Grajek, Jakub Szekiełda, Dominik Cywiński, Jakub Siejak, "[VCM] Poznań University of Technology Proposal C in response to CfP on Video Coding for Machines", ISO/IEC JTC 1/SC 29/WG 2 m60729, Online – October 2022.

[5] "Call for Proposals for Video Coding for Machines" wg2n00191

[6] ISO/IEC JTC1/SC29/WG5, "VVC Reference Model (VTM)"