Special Paper
방송공학회논문지 제25권 제7호, 2020년 12월 (JBE Vol. 25, No. 7, December 2020)
https://doi.org/10.5909/JBE.2020.25.7.1081
ISSN 2287-9137 (Online) ISSN 1226-7953 (Print)

Experiment on Intermediate Feature Coding for Object Detection and Segmentation

Min Hyuk Jeong[a)], Hoe-Yong Jin[a)], Sang-Kyun Kim[a)‡], Heekyung Lee[b)], Hyon-Gon Choo[b)], Hanshin Lim[b)], and Jeongil Seo[b)]

Abstract

With the recent development of deep learning, most computer vision-related tasks are being solved with deep learning-based network technologies such as CNN and RNN. Computer vision tasks such as object detection or object segmentation use intermediate features extracted from the same backbone such as Resnet or FPN for training and inference for object detection and segmentation. In this paper, an experiment was conducted to find out the compression efficiency and the effect of encoding on task inference performance when the features extracted in the intermediate stage of CNN are encoded. The feature map that combines the features of 256 channels into one image and the original image were encoded in HEVC to compare and analyze the inference performance for object detection and segmentation. Since the intermediate feature map encodes the five levels of feature maps (P2 to P6), the image size and resolution are increased compared to the original image. However, when the degree of compression is weakened, the use of feature maps yields similar or better inference results to the inference performance of the original image.

Keywords : Deep learning, intermediate features, video coding for machine, object detection, object segmentation

I. Introduction

Recently, research on image analysis using deep learning has been actively conducted due to the rapid progress of computer performance and the increasing demand for image recognition and automatic semantics extraction. Among them, the outstanding object extraction and segmentation performance of the Convolution Neural Network (CNN) is rapidly replacing traditional computer vision technology. Deep learning models can learn human faces to recognize human faces [1] or learn motion vectors in videos to determine similarity with other videos to determine whether they are duplicated [2]. Research on 3D object recognition for self-driving cars [3] or research on thermal infrared-based human detection [4] through model learning of IR images for application to self-driving car sensors is also drawing attention. Research using CNNs is being actively conducted in various computer vision fields such as estimating human movement in real-time [5], analyzing population density [6], and studying model learning for pedestrian detection [7].

a) MyongJi University
b) Electronics and Telecommunications Research Institute
‡ Corresponding Author : Sang-Kyun Kim
E-mail: goldmunt@gmail.com
Tel: +82-2-300-0637
ORCID: http://orcid.org/0000-0002-2359-8709
※ This work was supported by Institute of Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (No.2020-0-00011, Video Coding for Machine).
· Manuscript received November 5, 2020; Revised December 11, 2020; Accepted December 11,, 2020.

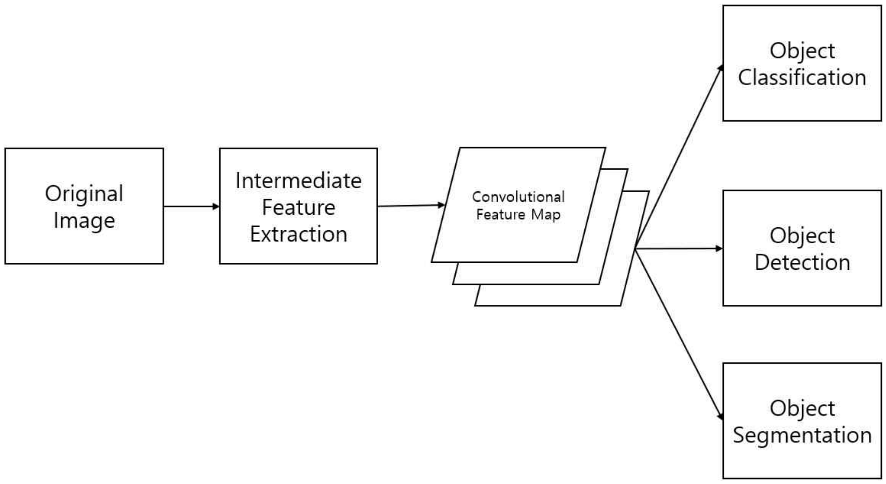


Fig. 1. The architecture of CNN based model

Figure 1 shows the process of CNN performing several computer vision tasks simultaneously. Object classification, object detection, and object segmentation all use intermediate feature maps extracted from the same backbone network for task training and inference. Since the same feature map is used even for different computer vision tasks, a method is needed for efficiently compressing and transmitting the feature maps to the post-processing network of each task.

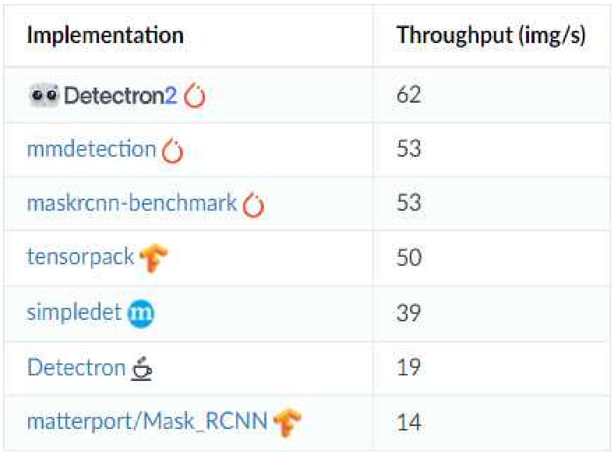
The feature map compression method for efficient feature map transmission is as follows. During the video search, the bit-rate reduction and search performance were maintained by coding local and global features for object detection and classification of images [8]. The study comparing the compression efficiency by compressing the feature map in GZIP, BZIP2, LZMA, and ZLIP methods [9] is insufficient in explaining the image compression performance by using simple text-based lossless compression tools. In the experiment [10], the feature map extracted for each backbone network layer of the YOLO9000 model was subjected to lossless and lossy compression and then retrained to compare the recognition rate and compression rate. There is a limitation in comparing only the feature maps extracted for each layer of the backbone network. The study on the memory efficiency during CNN Inference after performing Affine Transform Coding, PCA, Linear Quantization, and Variable Length Coding for feature maps, has a limitation in that it does not consider the efficiency of actual feature map data capacity compression [11].

Previous studies lacked the suggestion of how feature map compression affects object detection and segmentation performance, and the compression rate for each encoding condition (QP) compared to the original dataset. This paper provides experimental results on intermediate feature coding for CNN. During the inference process, when HEVC coding is performed on features extracted from the backbone and delivered to a subsequent network, the effect on the evaluation result was investigated. We evaluated the performance of object detection using Faster R-CNN X101-FPN and COCO datasets, and we evaluated the performance of object segmentation using the Mask R-CNN R50-FPN Cityscapes dataset.

The structure of the paper is as follows. Section 2 explains the structure of Detectron2, and Section 3 explains the sequence of experiments. Section 4 describes the experimental conditions, models, and datasets used, and Section 5 describes the experimental results. Finally, Section 6 concludes this paper.

II. Structure of Detectron2

Detectron2 is a platform for object detection and object segmentation based on the PyTorch library developed by FAIR (Facebook AI Research).



Implementation	Throughput (img/s)
Detectron2	62
mmdetection	53
maskrcnn-benchmark	53
tensorpack	50
simpledet	39
Detectron	19
matterport/Mask_RCNN	14

Fig. 2. Mask R-CNN's speed benchmark with various open sources

As shown in Figure 2, Detectron2 shows the best performance in terms of speed than Mask R-CNN implemented with other open sources. The reason Detectron2 can perform faster than other open sources is that python is well optimized, and parts that require a lot of computation are implemented in CUDA and C.

Detectron2 can be learned without implementing the training loop directly by the user, and it is modularized in each learning step, so you can easily change the desired part and learn. In addition, Detectron2 can load and apply weight models trained with popular CNN models. Through this, the inference can be performed without proceeding with learning.

Figure 3 is a picture of Faster R-CNN, which is a combination of R-CNN and FPN (Feature Pyramid Network). The red box of Feature 3 is FPN, which extracts feature maps for each step. Each of 256 channels is extracted from p2, which is 1/4 scale of the original image, to p6, which

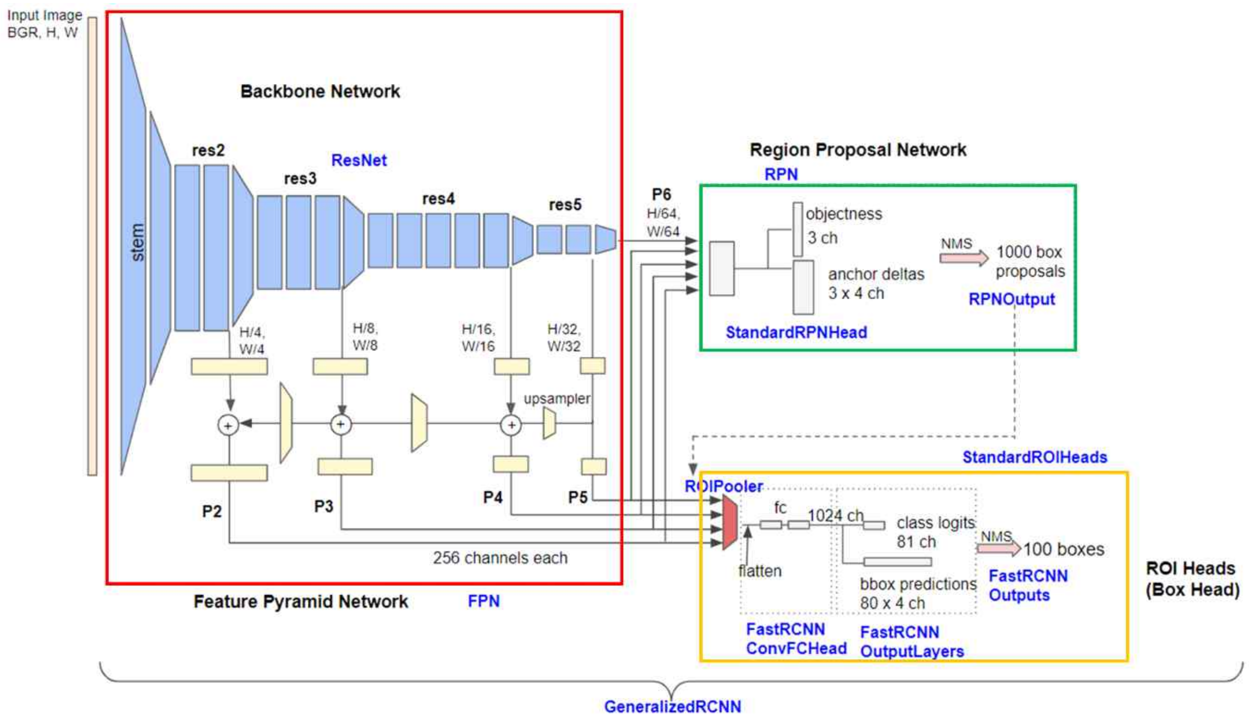


Fig. 3. The architecture of Base R-CNN FPN [15]

is 1/64 scale.

The feature maps of p2~p6 extracted by 256 channels are input to RPN (Region Proposal Network), the green box in Figure 3. RPN proposes an area that is likely to be an object from the input feature maps.

RoI Heads, which are yellow boxes, receive object regions proposed from RPN and feature maps extracted from FPN and perform object detection or object segmentation for the corresponding region.

Ⅲ. Processing Pipeline

Two experiments were conducted and the results were compared. One is the evaluation result of encoding and decoding with HEVC by packing the extracted 256-channel p2~p6 features into one 8bit grayscale image (feature frame) after feature extraction of the original images of the dataset. The other is the result of the evaluation by encoding and decoding input images with HEVC.

1. Process of encoding and decoding feature frames

The process is largely divided into two parts. It consists of a sender part that packs and encodes the feature frame and transmits it, and a receiver part that performs a task by decoding and unpacking the encoded feature frame (Fig. 4). The sender would extract features from the original images of the dataset and pack them into feature frames. Then, the sender saves the packed feature frame as an 8-bit grayscale PNG file and encodes the PNG file into the HEVC encoder with FFMPEG. At this time, BPP is calculated using the file size of the encoded feature frames with

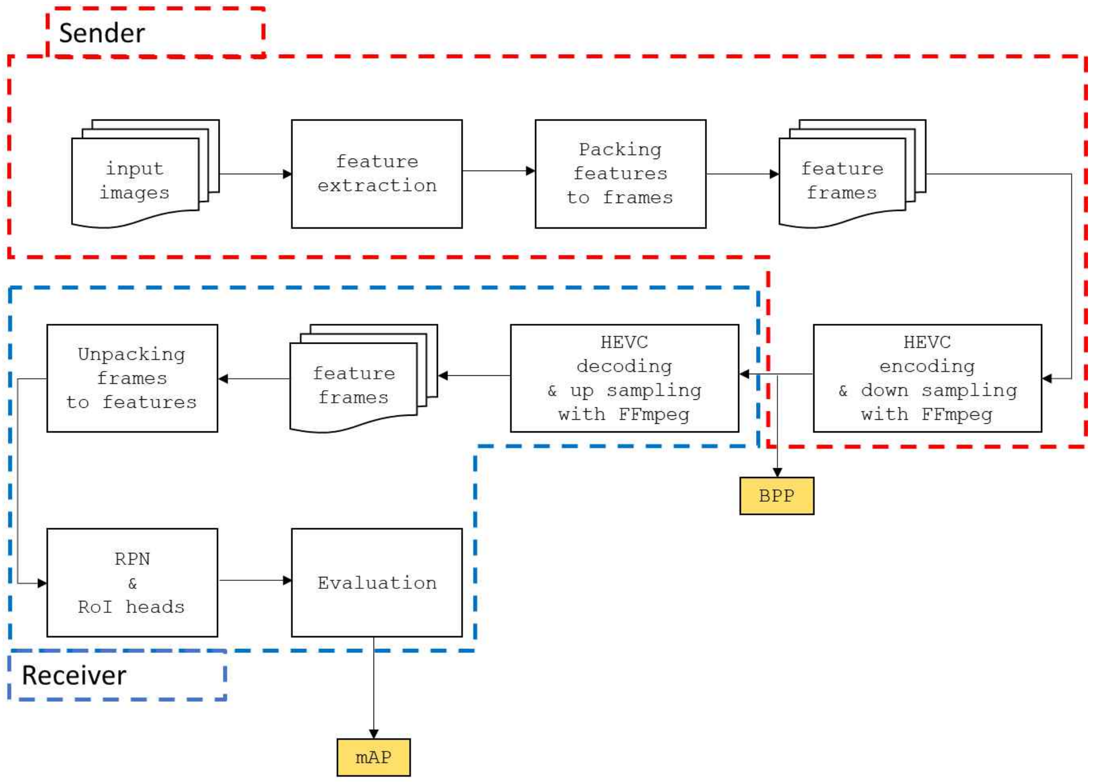

Sender
input images
feature extraction
Packing features to frames
feature frames
HEVC encoding & down sampling with FFmpeg
HEVC decoding & up sampling with FFmpeg
feature frames
Unpacking frames to features
BPP
RPN & RoI heads
Evaluation
Receiver
mAP

Fig. 4. The processing flow of encoding and decoding feature frames

respect to the resolution of the original input images. The receiver decodes and unpacks the encoded feature frames through FFMPEG and separates them into features (P2~P6). Instances are derived by entering the features obtained by unpacking as RPN and RoI head. The mAP (i.e., mean Average Precision) result is derived by evaluating through the derived instances.

2. Process of en/decoding input images

The original input images of the dataset are encoded with FFMPEG. At this time, BPP is calculated through the file size of the encoded images with respect to the resolution of the original images. The encoded images are decoded, input to the neural network, and evaluated through the derived instances to derive mAP (Fig. 5).

Ⅳ. Experiment Preparation

1. Coding environment

The coding environment is listed as follows:

- HEVC encoding/decoding: FFMPEG 4.2.2,
- Scaling resolution: 100%, 50%,
- QPs: 22, 27, 32, 37, 42, 47,
- Encoding pixel format: YUV420.

We use FFMPEG 4.4.2 for encoding and decoding the intermediate features and the input images. The scale resolutions employed are 100% and 50% of the original resolutions of the input images. When the features and the input images are compressed, five different QPs are given to evaluate the inference performance comparing to the degradation of the quantization. The encoded pixel format is YUV420.

2. Neural network environment

Table 1 shows a summary of the experiment environment related to neural networks. The neural network architecture is Faster R-CNN X101-FPN for object detection

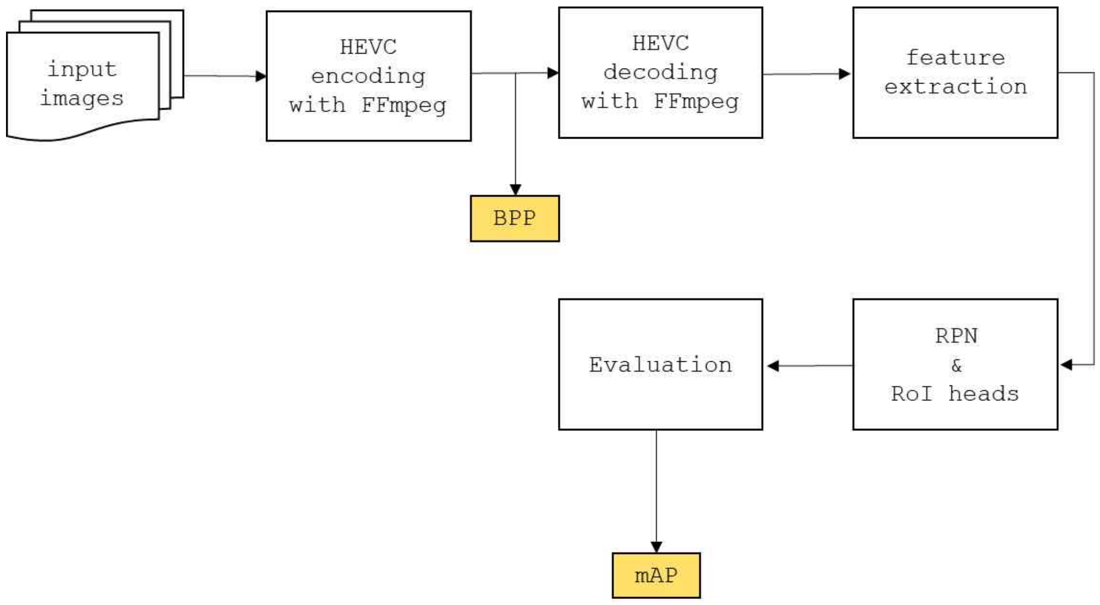


Fig. 5. The processing flow of encoding and decoding input images

Table 1. Summary of experiment environment regarding neural networks

Task	Object detection	Object segmentation
NN architecture name	Faster R-CNN X101-FPN (part of Facebook AI Research's Detectron2)	Mask R-CNN R50-FPN (part of Facebook AI Research's Detectron2)
NN architecture link	https://github.com/facebookresearch/detectron2	
Re-trained or modified in any way?	No	No
Dataset used by the original authors for training	COCO train 2017 (model_final_68b088.pkl)	COCO train 2017 (model_final_f10217.pkl)
Performance Metric	mAP(mean Average Precision)@0.5:0.95, BPP	
Dataset for evaluation: name and compression algorithm	COCO Val 2017 [13]	Cityscapes [14] (Cityscapes gtFine_val)
Is input data pre-processed to a fixed resolution?	default preprocessing in detectron2	
category(class)	All	

and Mask R-CNN R50-FPN for object segmentation, respectively. Both architectures were trained with the COCO Train 2017 set.

3. Dataset

For the COCO dataset [13], 2017 Val images [5K/1GB] are used, and all of the 5000 images are used. Some of the images in the COCO dataset have an odd number of widths or heights. To encode in YUV420 pixel format, both width and height must be even. In this case, the command line specified below is applied to make the width and height from odd numbers to even numbers.

- FFMPEG -i input -vf “pad=ceil(iw/2)*2:ceil(ih/2)*2“ output

Where the ‘iw’ denotes input width and the ‘ih’ denotes input height. As the image resolution is changed to an even number, the width and height values of the image information of the JSON file (instances_val2017.json) containing the ground truth information of the COCO Val 2017 dataset are also modified accordingly.

For the Cityscapes dataset, a validation set composed of 500 images (Frankfurt 267, Lindau 59, Munster 174) is used.

4. Image pre-processing of Detectron2

For evaluation, we utilized Detectron2 as a network platform [15]. Detectron2 has the default setting with minimum and maximum image input size. Input images smaller than the minimum input size (i.e., INPUT.MIN_SIZE_TEST) are upscaled while downscaling for input images larger than the maximum input size (i.e., INPUT.MAX_SIZE_TEST).

When Detectron2 loads the dataset, it normalizes the pixels of the image through pixel_mean and pixel_std in the following manner and converts them into signed floating number values.

- (pixel_value – pixel_mean) / pixel_std

Besides, padding is added so that the width and height are 64 times to extract p6 of the 1/64 scale without the scaling issue before inputting to the backbone.

5. Feature frame packing

The features (Fig. 6) of p2~p6 of 256 channels extracted from NN's backbone (FPN) are packed into one feature frame (Figs. 7 and 8).

To save the feature frame in PNG format, the pixel value is converted to an 8-bit unsigned integer. The following equation is used for rescaling.

$$\mathrm{rescaled\_pixel} = 255 * \frac{\mathrm{cur\_pixel} + \mathrm{abs}(\mathrm{min\_pixel})}{\mathrm{max\_pixel} + \mathrm{abs}(\mathrm{min\_pixel})}$$

In the equation, max_pixel is the largest pixel value of

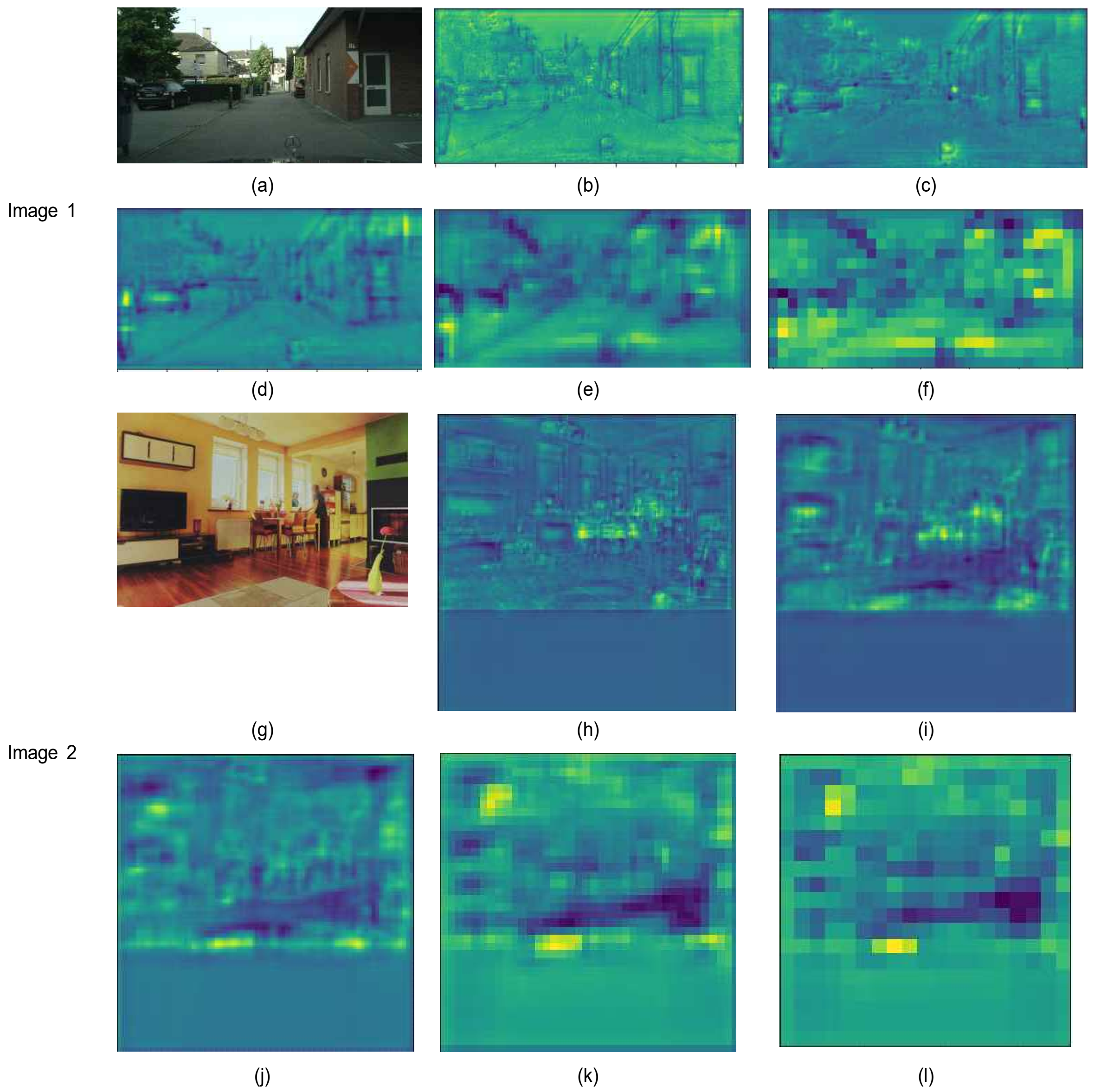


Fig. 6. original input images: (a)(g), feature images: (b)(h) p2, (c)(i) p3, (d)(j) p4, (e)(k) p5, (f)(l) p6

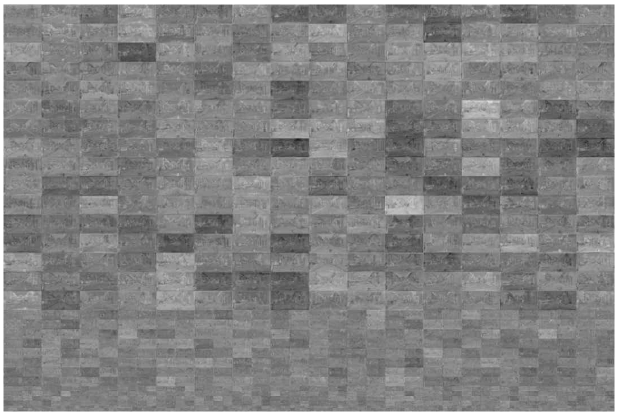


Fig. 7. Feature frame of Image 1 (Fig. 6)

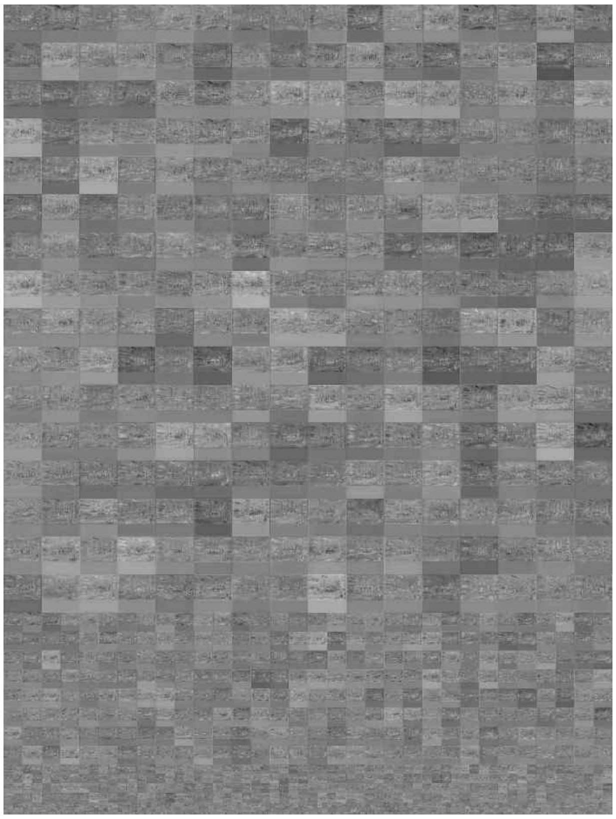
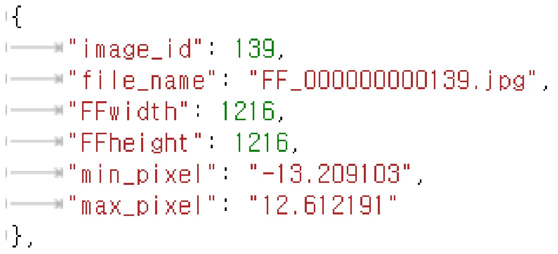


Fig. 8. Feature frame of Image 2 (Fig. 6)

the feature frame and min_pixel is the smallest pixel value. For example, if the current min_pixel value is -13.2 and the max_pixel value is 12.6, the smallest pixel value is changed to 0 and the largest pixel value is changed to 25.8 through the first equation. Here, multiplying all pixels by 9.88, which is 255/25.8 through the second equation, converts the pixel value to a value between 0 and 255. After converting to a value between 0 and 255, typecasting was performed into an unsigned integer type. After conversion, it was saved as an 8-bit grayscale image. When unpacking the feature frame, transfer the min_pixel and max_pixel values together with image information in a JSON file so that the min_pixel and max_pixel values can be used in reverse order (Fig. 9).



```
{
	"image_id": 139,
	"file_name": "FF_000000000139.jpg",
	"FFwidth": 1216,
	"FFheight": 1216,
	"min_pixel": "-13.209103",
	"max_pixel": "12.612191"
},
```

Fig. 9. Example of JSON data

6. HEVC encoding and decoding

For HEVC coding, FFMPEG 4.2.2 version is used. For encoding, the following command line with parameters is being used.

- FFMPEG -f image2 -i input -pix_fmt yuv420p -c:v libx265 -x265-params qp=QP -f rawvideo output

For feature frames, the following command line with parameters is being used.

- FFMPEG -f image2 -i input -pix_fmt yuv420p -c:v libx265 -x265-params qp=QP -vf “(NW:NH)” -f rawvideo output

where, QPs used are 22, 27, 32, 37, 42, and 47. NW and NH denote width and hight of target image sizes using the following equations.

- 100%: pad=ceil(iw/2)*2:ceil(ih/2)*2
- 50%: scale=ceil(iw/4)*2:ceil(ih/4)*2

The following command line with parameters is being used for dataset input image decoding.

- FFMPEG – i input -pix_fmt rgb24 -f image2 output

The following command line with parameters is being used for feature frame decoding.

- FFMPEG -i input -f image2 -pix_fmt gray -vf (NW:NH) output

For feature frames downscaling to 50%, upscaling was performed to the original resolution, and NW and NH were calculated as follows.

- 50%: scale=iw*2:ih*2

7. Feature frame unpacking

To read feature frames encoded by HEVC and use them as inputs for the RPN and RoI heads, we unpack them into a feature dictionary of p2~p6 of 256 channels. At this time, the pixel values converted to values between 0 and 255 to be saved in the PNG image format must be converted back to signed floating numbers. Based on the JSON data created in the feature frame packing step, the calculation formula used in the feature frame packing is inverted and converted into a signed floating number form.

$$\mathrm{rescaled\_pixel} = \frac{cur\_pixel * (\mathrm{max\_pixel} + \mathrm{abs}(\mathrm{min\_}pixel)}{255} - \mathrm{abs}(\mathrm{min\_pixel})$$

For example, if the current min_pixel value is -13.2 and the max_pixel value is 12.6, all pixel values are divided by 9.88, the value of 255 / 25.8 through the first equation. If so, the current smallest pixel value is 0, and the largest pixel value is 25.8. Here, by subtracting 13.2, the absolute value of min_pixel, from all pixel values through the second equation, the smallest pixel value is -13.2 and the largest pixel value is 12.6.

8. BPP calculation

The BPP for images and feature frames without encoding/decoding is calculated as follows.

- Input images: File size of original input images / Resolution of original input images

The BPP calculation for the encoded/decoded images and feature frames was performed as follows.

- Input images: File size of encoded input images / Resolution of original input images
- Feature frames: File size of encoded feature frames / Resolution of original input images

9. Performance measurement

Performance results were measured through mAP and BPP derived through the evaluator. When evaluating COCO and Cityscapes, each evaluator provided by Detectron2 is used.

Ⅴ. Experiment Results

The experiment compared the results of HEVC encoding/decoding on the original dataset images with 100% resolution and the result of scaling the feature frames to 100% and 50% resolution when HEVC encoding/decoding. In addition, the result of the image and feature frame without encoding/decoding is included.

“No enc” in the tables and figures indicates the images or feature frames are not encoded. The coded images or feature frames with HEVC are marked with QPs. The results with the highest mAP were highlighted in yellow in the tables.

1. Object detection

Table 2 shows the object detection performance measured by mAP and BPP for different QPs and resolutions on feature frames. Table 3 shows the object detection performance measured by mAP and BPP for different QPs and resolutions on original images.

Table 2. Experimental results on feature frames of the COCO dataset for object detection

Resolution	QP	mAP	BPP
100%	No enc.	43.058	
	22	43.051	73.78952725
	27	43.099	41.23695566
	32	42.898	20.71645289
	37	42.177	9.393729824
	42	39.749	4.002137245
	47	30.022	1.579919992
50%	No enc.	32.983	
	22	34.674	23.57866023
	27	34.561	13.30736244
	32	34.013	6.637972284
	37	31.570	2.988037762
	42	22.532	1.278723157
	47	5.597	0.506873677

Table 3. Experimental results on original images of the COCO dataset for object detection

Resolution	QP	mAP	BPP
100%	No enc.	43.047	4.762082979
	22	40.739	2.161297571
	27	40.325	1.445341381
	32	38.925	0.897053358
	37	35.843	0.53217704
	42	29.604	0.308422173
	47	18.781	0.171853764

The mAPs for the COCO original image and the unencoded feature frame with a 100% resolution were 43.047 and 43.058 respectively. The mAPs for QP22 and the QP27 were 43.051 and 43.099, showing higher mAP results than the original image. The mAPs with a 50% resolution were 34.674, 34.561, and 34.031 for QP22, QP27, and QP32, respectively, which are higher than 32.983 of 50% unencoded image. However, both the BPP 41.237 of QP27 with a 100% resolution and the BPP 23.579 of Q22

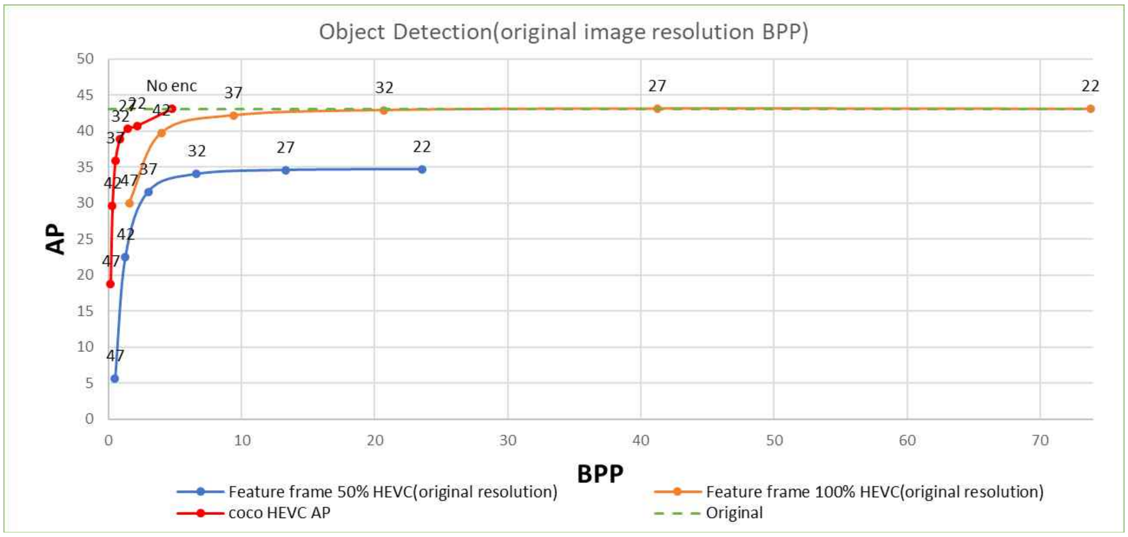


Fig. 10. Experimental results on the COCO dataset for object detection

with a 50% resolution is too much higher than the BPP 4.762 of the original unencoded image. Figure 10 shows the graphical trend of the experiment results in the object detection task. The orange line in Figure 10, “Feature Frame 100% HEVC (original resolution)” is an mAP/BPP for the original image resolution of the coco dataset of the 100% resolution feature frame in Table 2. The blue line “Feature Frame 50% HEVC (original resolution)” is an mAP/BPP graph for the original image resolution of the coco dataset of a 50% resolution feature frame. The red line “coco HEVC AP” is an mAP/BPP graph of the original image in Table3.

2. Object segmentation

Table 4 shows the object segmentation performance measured by mAP and BPP for different QPs and resolutions on feature frames. Table 5 shows the object segmentation performance measured by mAP and BPP for different QPs and resolutions on original images.

The mAPs for the cityscape original images and the unencoded feature frames with a 100% resolution were36.480 and 36.310, respectively. The mAPs with a 50% resolution

Table 4. Experimental results on feature frames of Cityscapes dataset for object segmentation

Resolution	QP	mAP	BPP
100%	No enc.	36.310	
	22	36.081	20.1977117
	27	35.817	10.9292068
	32	35.214	5.25550312
	37	32.609	2.28282748
	42	25.878	0.94584658
	47	9.682	0.35711496
50%	No enc.	33.923	
	22	33.671	6.01710119
	27	32.485	3.25488912
	32	29.477	1.56421667
	37	20.031	0.68204609
	42	4.273	0.28202600
	47	0.047	0.10587720

Table 5. Experimental results on original images of Cityscapes dataset for object segmentation

Resolution	QP	mAP	BPP
100%	No enc.	36.480	9.01341506
	22	32.883	0.27173653
	27	28.582	0.15529182
	32	21.099	0.09025314
	37	12.946	0.05344588
	42	4.509	0.03177255
	47	1.214	0.02095923

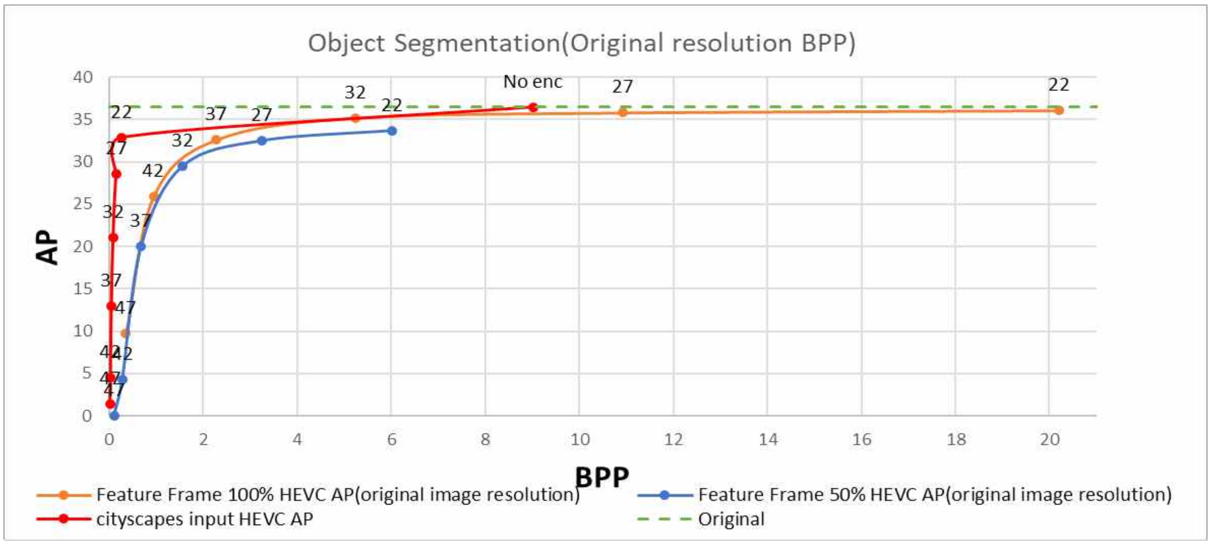


Fig. 11. Experimental results on Cityscapes dataset for object segmentation

of unencoded feature frames were 33.923. Feature frames do not achieve any mAP performance gains over the original images. Again, the BPPs from the feature frames are much higher than the ones from the original images. Figure 11 shows the graphical trend of the experiment results in the object segmentation task. The orange line in Figure 11, "Feature Frame 100% HEVC (original resolution)" is an mAP/BPP for the resolution of the original image of the cityscapes dataset of the 100% resolution feature frame in Table 4. The blue line "Feature Frame 50% HEVC (original resolution)" is an mAP/BPP graph for the resolution of the original image of the cityscapes dataset of the 50% resolution feature frame. The red line "cityscapes input HEVC AP" is an mAP/BPP graph of the HEVC coding result of the original input image in Table 5.

Ⅵ. Conclusion

We reported the experimental results of two tasks for intermediate feature coding. All the detailed experiment processes and settings are presented. Experimental results show that the mAPs of encoded feature frames can be similar or higher than those of the original input images. However, there is a disadvantage in that the resolution of the generated feature frame increases compared to the resolution of the input image. We will further study the high compression method of feature frames while minimizing mAP performance degradation for various vision tasks.

References

[1] Huaizu Jiang, Erik Learned-Miller, "Face Detection with the Faster R-CNN", IEEE 12th International Conference on Automatic Face & Gesture Recognition, 2017.
[2] Yuanyuan Yang, Yixiong Zou, Qingsheng Yuan, Yaowei Wang, Yonghong Tian, "Fast Compressed Domain Copy Detection with Motion Vector Imaging", IEEE Conference on Multimedia Information Processing and Retrieval, 2018.
[3] Peiliang Li, Xiaozhi Chen, Shaojie Shen, "Stereo R-CNN based 3D Object Detection for Autonomous Driving", Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7644-7652, 2019.
[4] Christian herrmann, Miriam Ruf, Jürgen Beyerer, "CNN-based thermal infrared person detection by domain adaptation", SPIE 10643 Autonomous Systems: Sensors, Vehicles, Security, and the Internet of Everything, May 2018.
[5] Zhe Cao, Tomas Simon, Shih-En Wei, Yaser Sheikh, "Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields", Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7291-7299, 2017.
[6] Vishiwanath A. Sindagi, Vishal M. Patel, "CNN-based Cascaded Multi-task Learning of High-level Prior and Density Estimation for Crowd Counting", 14th IEEE International Conference on Advanced Video and Signal Based Surveillance, 2017.
[7] Hui Zhang, Shurong Ning, Shuo Yang, Yu Du, Yonghua Zhang, Chen Du, "Pedestrian detection method based on Faster R-CNN", 13th IEEE International Conference on Computational Intelligence and Security, 2017.
[8] Lin Ding, Yonghong Tian, Hongfei Fan, Changhuai Chen, Tiejun Huang, "Joint Coding of Local and Global Deep Features in Videos for Visual Search", IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL 29, January 2020.
[9] Zhuo Chen, Weisi Lin, Shiqi Wang, Lingyu Duan, Alex C. Kot, "Intermediate Deep Feature Compression: the Next Battlefield of Intelligent Sensing", arXiv:1809.06196, September 2018.
[10] Hyomin Choi, Ivan V. Bajic, "Deep Feature Compression for Collaborative Object Detection", 25th IEEE International Conference on Image Processing, 2018.
[11] Brain Chmiel, Chaim Baskin, Evgenii heltonozhskii, Ron Banner, Yevgeny Yermolin, Alex Karbachevsky, Alex M. Bronstein, Avi Mendelson, "Feature Map Transform Coding for Energy-Efficient CNN Inference", International Joint Conference on Neural Network, 2020
[12] N19507, "Draft Evaluation Framework for Video Coding for Machines," 131st MPEG Online Meeting.
[13] COCO: Common Objects in Context, https://cocodataset.org.
[14] Cityscapes dataset, https://www.cityscapes-dataset.com/.
[15] Detectron2, https://github.com/facebookresearch/detectron2.

Introduction Authors

Min Hyuk Jeong

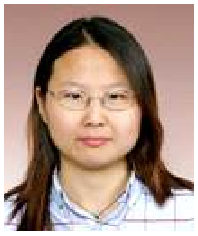
- 2009. ~ 2016. : Dept. of Computer Engineering, Myongji University, B.S
- 2016. 02. ~ 2018. 08. : Dept. of Computer Engineering, Myongji University, M.S
- 2019. 02. ~ Current : Dept. of Computer Engineering, Myongji University, Ph.D
- ORCID : https://orcid.org/0000-0001-6487-9219
- Research interests : Internet of Things, Virtual Reality, 4D media, Deep Learning

Hoe-Yong Jin

- 2013. 03. ~ 2019. 02. : Dept. of Computer Engineering, Myongji University B.S
- 2019. 03. ~ Current : Dept. of Data Technology, Myongji University, M.S
- ORCID : https://orcid.org/0000-0003-3749-5337
- Research interests : 4D media, Blockchain, Internet of Things and VR

Sang-Kyun Kim

- 1997. : Computer Science, Univ. of Iowa, B.S.(1991), M.S(1995), Ph.D(1997)
- 1997. 03. ~ 2007. 02. : Professional Researcher, Multimedia Lab. of Samsung Advanced Institute of Technology
- 2007. 03. ~ 2016. 02. : Professor of Computer Engineering, Myongji University
- 2016. 03. ~ Current : Professor of Software Convergent, Myongji University
- ORCID : https://orcid.org/0000-0002-2359-8709
- Research interests : Digital Content(image, video and audio) analysis and management, 4D media, Blockchain, VR, Internet of Things and multimedia standardization



Heekyung Lee

- 1999. 02 : Computer Engineering, Yeungnam University (B.S)
- 2002. 02 : Information & Communication Engineering, KAIST-ICC (M.S)
- 2002년 ~ Current : Senior Researcher, ETRI, Daejeon, Korea
- ORCID : https://orcid.org/0000-0002-1502-561X
- Research interests : Digital Broadcasting & Metadata, HCI, Gaze Tracking, VR/AR/MR, Deep-learning based stitching, Unsupervised learning based quality measure

Hyon-Gon Choo

- Senior Researcher, ETRI, Daejeon, Korea(2005~)
- Director of the Digital Holography Section(2015~2017)
- Visiting Researcher, Warsaw University of Technology, Poland(2017-2018)
- ORCID : https://orcid.org/0000-0002-0742-5429
- Research interests : Multimedia Signal Processing with emphasis on Digital Video Processing Techniques and Applications, including 3D imaging and holography/3D depth imaging/3D broadcasting system/Computer vision

Introduction Authors

Hanshin Lim

- 2004.02 : Dept. of Electronic and Electrocal Engineering (Minor: Mathematics), Yonsei University (B.S)
- 2006.02 : Dept. of Electronic and Electrocal Engineering, KAIST (M.S)
- 2007.09 ~ 2007.12 : Visiting Researcher, TU Berlin
- 2014.02 : Dept. of Electronic and Electrocal Engineering, KAIST (Ph.D.)
- 2014.03 ~ Current : Senior Researcher, ETRI
- ORCID : https://orcid.org/0000-0003-4829-2893
- Research interests : 2D/3D Image Processing, Computer Vision, 3D Reconstruction and Modeling, VR/AR Technology

Jeongil Seo

- 1994. 02 : Dept. of Electronic Engineering, Kyungpook University (B.S)
- 1996. 02 : Dept. of Electronic Engineering, Kyungpook University (M.S)
- 2005. 08 : Dept. of Electronic Engineering, Kyungpook University (Ph.D.)
- 1998. 02 ~ 2000.10 : Chief Researcher, LG Semiconductor
- 2010. 08 ~ 2011.07 : Visiting Researcher, Southampton University, ISVR, England
- 2000.11 ~ Current : Director of Immersive Media Section, ETRI,
- ORCID : http://orcid.org/0000-0001-5131-0939
- Research interests : Audio Signal Processing, Realistic Sound, Digital broadcasting, Multimedia Standardization