Supplementary Material to Neural Video Compression with Feature Modulation

Jiahao Li, Bin Li, Yan Lu
Microsoft Research Asia
{li.jiahao, libin, yanlu}@microsoft.com

This document provides the supplementary material to our proposed neural video codec (NVC), i.e. DCVC-FM.

1. Test Settings

For a thorough comparative analysis, we compare the NVCs and traditional codecs in both YUV420 and RGB colorspaces.

YUV420 colorspace. Most traditional codecs and practical applications mainly adopt YUV420 colorspace as input and output, and are optimized in this colorspace. Thus, the comparison between NVC and traditional codec in YUV420 colorspace is quite important for evaluating the development progress of NVC. For traditional codec, HM [2], VTM [3], and ECM [1] are tested, where HM, VTM, ECM are the reference software of H.265, H.266, and the under-developing next generation traditional codec, respectively. For the three traditional codecs, *encoder_lowdelay_main10.cfg*, *encoder_lowdelay_vtm.cfg*, and *encoder_lowdelay_ecm.cfg* config files are used, respectively. The parameters for each video are as:

- -c {*config file name*}
  --InputFile={*input video name*}
  --InputBitDepth=8
  --OutputBitDepth=8
  --OutputBitDepthC=8
  --FrameRate={*frame rate*}
  --DecodingRefreshType=2
  --FramesToBeEncoded={*frame number*}
  --SourceWidth={*width*}
  --SourceHeight={*height*}
  --IntraPeriod={*intra period*}
  --QP={*qp*}
  --Level=6.2
  --BitstreamFile={*bitstream file name*}

RGB colorspace. As the raw formats of all testsets are in YUV420 colorspace. Thus, to test RGB video, we need to convert them from YUV420 to RGB colorspace. We follow JPEG AI [4, 5] and [6], and use BT.709 to convert the raw YUV420 video to RGB video. This is because using BT.709 obtains higher compression ratio under the similar visual quality when compared with the commonly-used BT.601. [6] shows, when traditional codecs test RGB videos, using 10-bit YUV444 as the internal colorspace achieves better compression ratio than directly using RGB, although the final distortion is measured in RGB. So we also follow this setting. For HM, VTM, and ECM, *encoder_lowdelay_main_rext.cfg*, *encoder_lowdelay_vtm.cfg*, and *encoder_lowdelay_ecm.cfg* config files are used, respectively. The parameters for each video are as:

- -c {*config file name*}
  --InputFile={*input file name*}
  --InputBitDepth=10
  --OutputBitDepth=10
  --OutputBitDepthC=10
  --InputChromaFormat=444
  --FrameRate={*frame rate*}
  --DecodingRefreshType=2
  --FramesToBeEncoded={*frame number*}
  --SourceWidth={*width*}
  --SourceHeight={*height*}
  --IntraPeriod={*intra period*}
  --QP={*qp*}
  --Level=6.2
  --BitstreamFile={*bitstream file name*}

It is noted that, for both YUV420 and RGB colorspaces, all coding tools and reference structure of traditional codecs use their best settings to represent their best compression ratio.

2. Temporal Feature Modulation

In this paper, we specially design the feature refresh mechanism to modulate the temporal feature. It will improve the effectiveness of feature propagation. The default refresh period is set as 32. Here we test different refresh period settings, and the corresponding comparisons are in Table 1, where the refresh period 0 means disabling the feature refresh. From this table, we can see that the bitrate saving initially increases with the refresh period size. When the refresh period size is larger than 32 (i.e. 64 and 96), the performance begins to decay. This phenomenon shows that both too small or too large refresh period sizes

Table 1. BD-Rate (%) of using different feature refresh periods.

Refresh period	0	8	16	32	64	96
BD-Rate	0	–7.2	–17.1	–20.0	–17.9	–14.8

are not proper. If period size is too small, the addition bitrate cost is too heavy because the frame with feature refresh will not only have better quality but also have a larger bitrate cost. Conversely, an excessively large period size can lead to the propagation of features contaminated by the accumulated errors or containing higher amounts of uncorrelated information. So currently the refresh period 32 is a good trade-off. Nevertheless, it should be noted that the best refresh period size may vary across different videos. A content-adaptive approach to determine the refresh period size would likely yield better results. We will investigate it in the future.

3. Rate Control

We implement a simple rate control algorithm to just showcase the feasibility by adjusting the quantization parameter q_t values in the model. Because of the bitrate fluctuation among frames, we only adjust the quantization parameter q_t at even frames. The rate control algorithm is provided in Algorithm 1. It should be noted that this algorithm is a simple implementation to demonstrate the feasibility of rate control. More advanced rate control algorithms can be designed based on our codec and we will investigate it in the future.

4. Reimplemented *grid_sample*

The reimplemented *grid_sample* function can be found in our release codes. We also compare the error ratio for the randomly generated feature and motion vector. The error ratio is as high as 16.149% if we directly feed the 16-bit tensor value to the default *grid_sample* function. By contrast, the error ratio is 0.026% if using our reimplemented and improved *grid_sample* function.

5. Rate-Distortion Curves

In this document, we show the rate-distortion (RD) curves of all datasets with testing all frames and using intra period –1 setting. Fig. 1 shows the results of HEVC B, C, and D datasets, and Fig. 2 shows those of HEVC E, UVG, and MCL-JCV datasets. In these two figures, we also compare the relatively low and high quality regions, respectively. From these comparisons, we can see that previous SOTA NVC DCVC-DC [6] has quite limited quality range. By contrast, our DCVC-FM has much wider range, and achieves the best RD performance over all previous codecs for many cases.

Algorithm 1 Rate Control Algorithm

```
Input:
    $cbs$: current buffer size (set as 0 for the first frame)
    $tbs$: target buffer size (set as 0 for the first frame)
    $q$: current q value (set as 32 for the first frame)
    $cfs$: current frame size
    $afs$: average frame size
    $fidx$: current frame index
Output
    $cbs$: updated buffer size for the next frame
    $tbs$: updated target buffer size for the next frame
    $q$: updated q value for the next frame

  $cbs += cfs$
  $cbs -= afs$
  if $fidx \mod 2 == 1$ then
      return $cbs$, $tbs$, $q$
  end if
  $buff\_diff = cbs - tbs$
  $tbs = cbs * 0.95$
  if $buff\_diff > 0$ then
      if $cbs > 10 * cfs$ then
          $q -= 12$
      else if $cbs > 5 * cfs$ then
          $q -= 6$
      else if $cbs > 2 * cfs$ then
          $q -= 2$
      else if $buff\_diff > 0.5 * cfs \ \& \ cbs > -cfs$ then
          $q -= 1$
      end if
  else if $buff\_diff < 0$ then
      if $cbs < -10 * cfs$ then
          $q += 12$
      else if $cbs < -5 * cfs$ then
          $q += 6$
      else if $cbs < -2 * cfs$ then
          $q += 2$
      else if $buff\_diff < -0.5 * cfs \ \& \ cbs < cfs$ then
          $q += 1$
      end if
  end if
  $q = clip(0, 63, q)$
  return $cbs$, $tbs$, $q$
```

But as shown in Fig. 2, our DCVC-FM cannot surpass ECM on MCL-JCV dataset. MCL-JCV includes screen content videos. We find our codec is not good at screen content. It is because our training dataset Vimeo is a natural content dataset. In addition, our codec also performs worse on video with lots of noise. Currently, it is quite hard for neural codec to code the random noise in source video as the probability of noise data is difficult to predict accu-

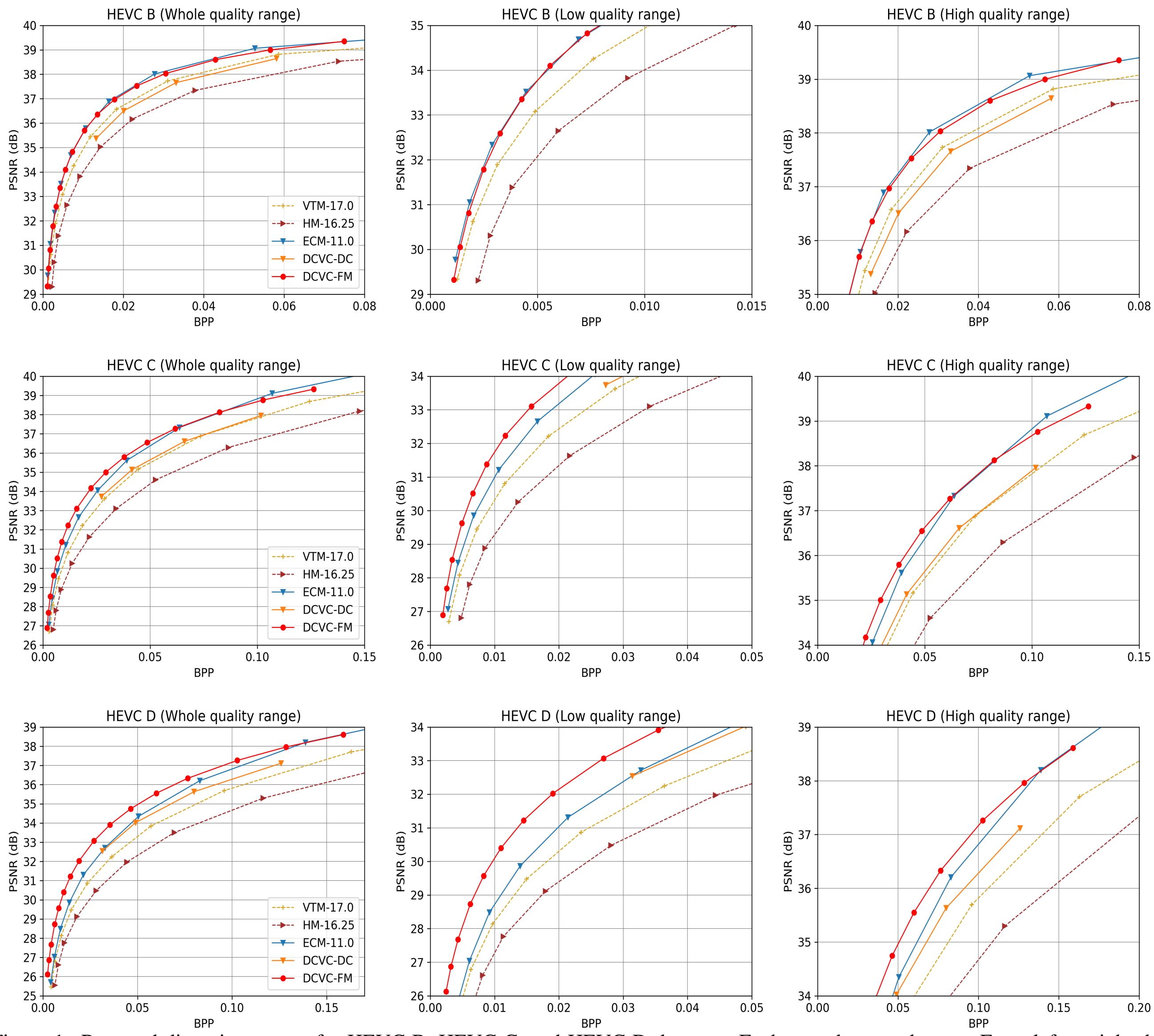

Figure 1. Rate and distortion curves for HEVC B, HEVC C, and HEVC D datasets. Each row shows a dataset. From left to right the figures are overall quality range, relatively low quality range and relatively high quality range, respectively. The comparison is in YUV420 colorspace. All frames with intra-period = –1.

rately. In the future, we will improve our codec on these videos.

6. Smooth Quality Adjustment in Single Model

For convenience, we test 16 RD points when comparing our DCVC-FM with other codecs. Actually, our NVC can support 64 different quality levels in single model. We test all these RD points, as shown in Fig. 3. From these figures, we can see that our DCVC-FM can achieve very smooth quality adjustment in single model, and there is no any outlier in the RD curves. This is also the prerequisite of achieving precise rate control.

7. Visual Comparison

In this section, we offer visual comparisons to illustrate the superior performance of our DCVC-FM. Four examples are presented in Fig. 4. These examples demonstrate that our DCVC-FM is capable of reconstructing textures with greater clarity, without incurring additional bitrate costs, when compared with traditional codec ECM and previous SOTA NVC DCVC-DC.

8. Traditional Codec Using B Frame Config

Currently our neural codec focuses on low-delay scenario, so the traditional codec uses low-delay-B (LDB) setting for

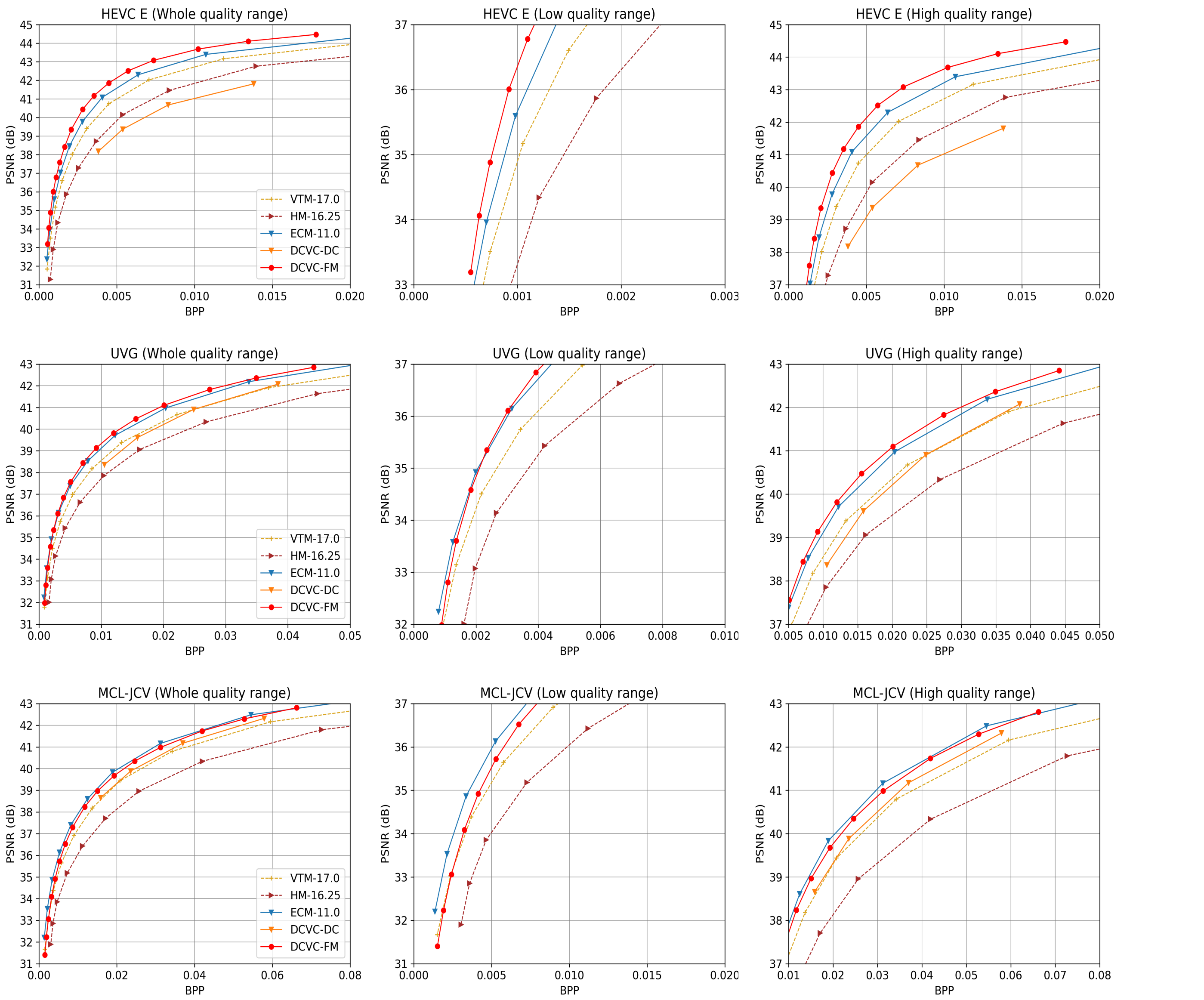

Figure 2. Rate and distortion curves for HEVC E, UVG, and MCL-JCV datasets. Each row shows a dataset. From left to right the figures are overall quality range, relatively low quality range and relatively high quality range, respectively. The comparison is in YUV420 colorspace. All frames with intra-period = –1.

fair comparison in the main paper. Actually, we also already tested the hierarchical-B (HieB) setting (the random-access config with intra-period = –1, where the low-delay requirement is broken) for ECM-5.0 under 96 frames, as shown in Table 2. The compression ratio gap between LDB and HieB is about 26%, consistent with the number reported in JCTVC-K0279 (21% on average for HEVC). Designing a neural codec which surpasses the best traditional codec in HieB setting will be future work.

Table 2. BD-Rate (%) comparison (YUV420, 96 frames, intra-period = –1).

	UVG	MCL-JCV	HEVC B	HEVC C	HEVC D	HEVC E	Average
VTM-17.0 (LDB)	0.0	0.0	0.0	0.0	0.0	0.0	0.0
ECM-5.0 (LDB)	–13.3	–16.4	–14.6	–15.6	–14.1	–12.6	–14.4
ECM-5.0 (HieB)	–40.3	–40.5	–42.3	–39.1	–39.3	–39.4	–40.2
DCVC-FM (Low-delay)	–25.4	–11.6	–17.1	–24.4	–41.5	–31.6	–25.3

References

[1] ECM. https://vcgit.hhi.fraunhofer.de/ecm/ECM. 1

[2] HM. https://vcgit.hhi.fraunhofer.de/jvet/HM/. 1

[3] VTM. https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware_VTM/. 1

[4] E. Alshina, J. Ascenso, T. Ebrahimi, F. Pereira, and T. Richter. [AHG 11] Brief information about JPEG AI CfP status. In

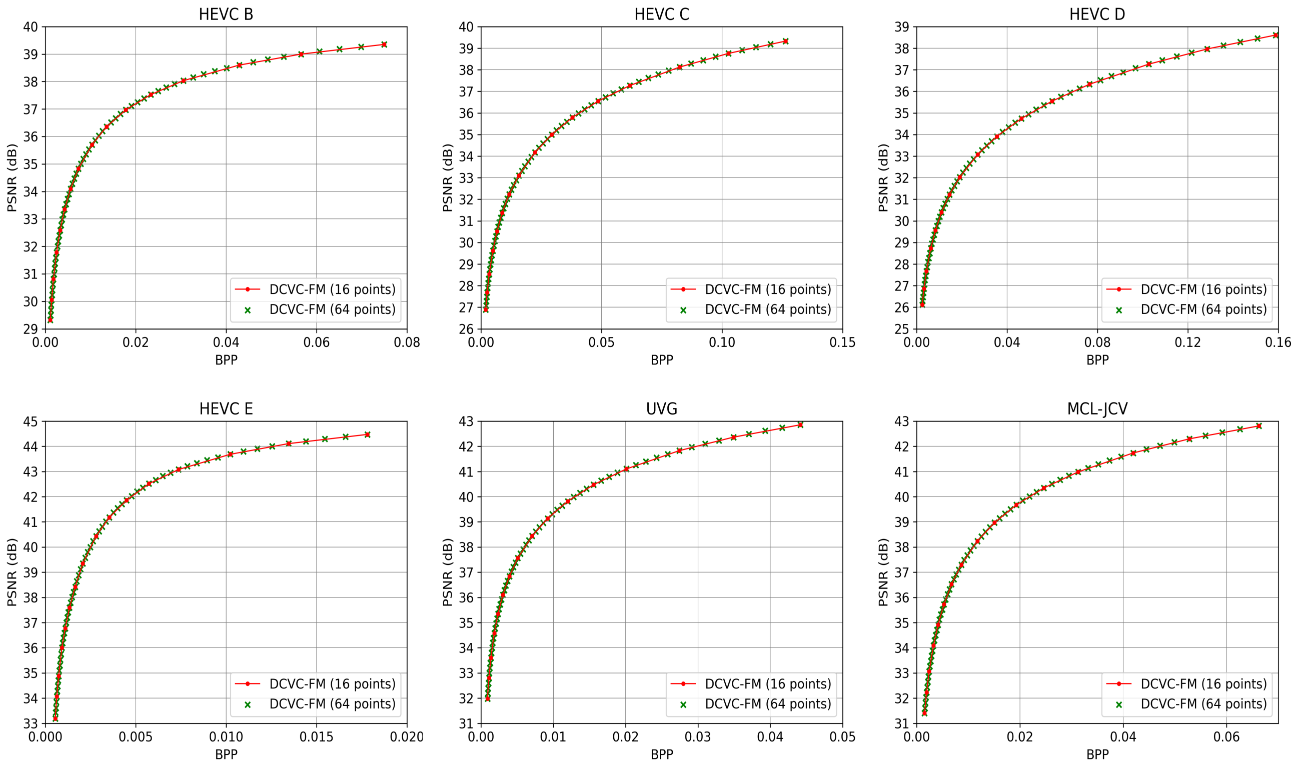

Figure 3. Smooth quality adjustment in single model. Our DCVC-FM supports 64 different quality levels.

JVET-AA0047, 2022. 1

[5] Anchors · JPEG-AI MMSP Challenge. Anchors · JPEG-AI MMSP Challenge. `https://jpegai.github.io/7-anchors/`. 1

[6] Jiahao Li, Bin Li, and Yan Lu. Neural video compression with diverse contexts. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, Canada, June 18-22, 2023*, 2023. 1, 2

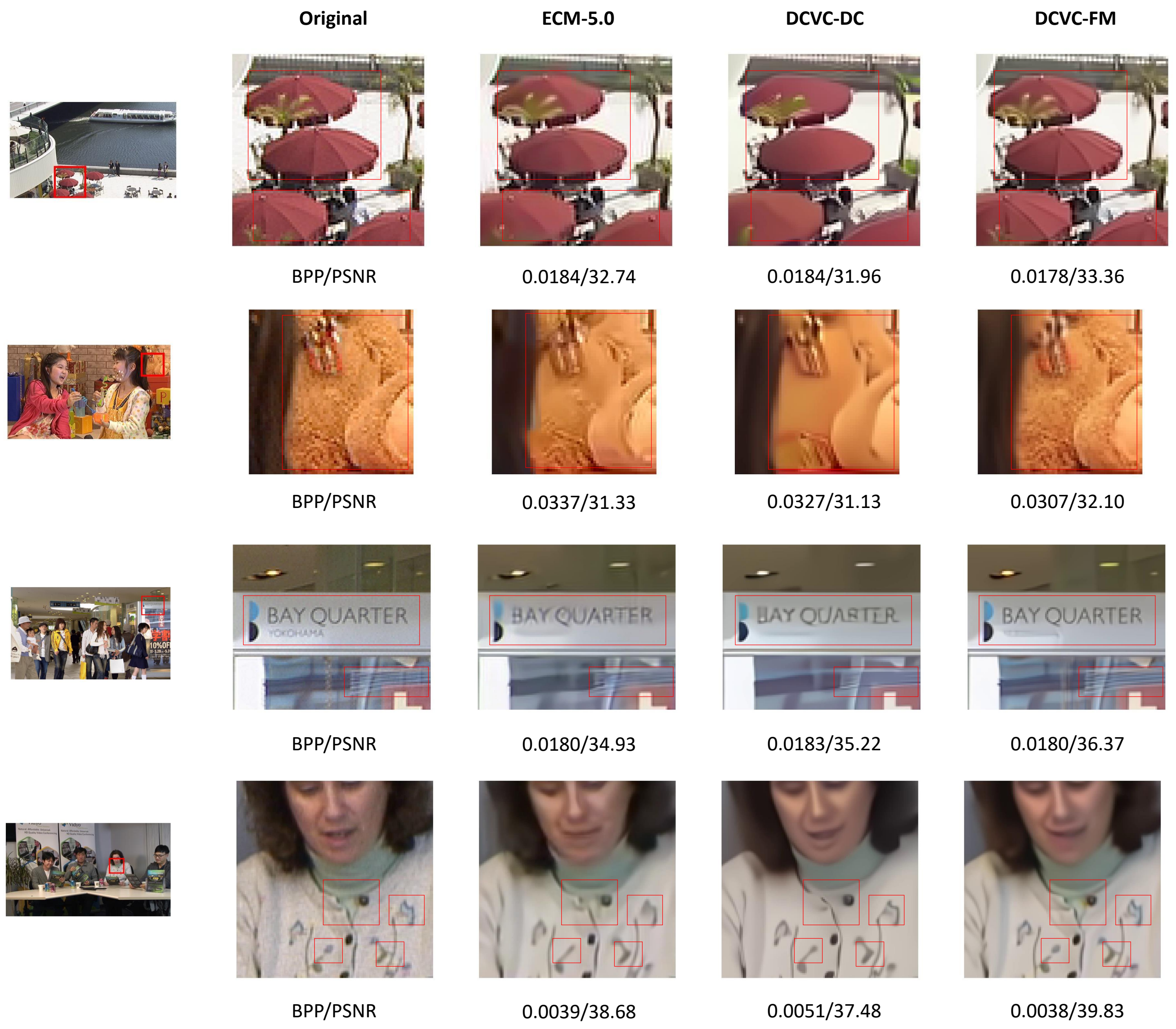

Figure 4. Visual comparisons.