Supplemental Material

Zhimeng Huang[1][0000−0001−8026−9349], Chuanmin Jia[2][0000−0002−7418−6245], Shanshe Wang[1][0000−0002−7665−7434], and Siwei Ma[1][0000−0002−2731−5403]

[1] National Engineering Research Center of Visual Technology, Peking University, Beijing 100871, China
[2] Wangxuan Institute of Computer Technology, Peking University, Beijing 100871, China

1 Supplemental Experimental Configurations

In this section, the source codes, instructions and configurations of codecs and VOS methods are provided.

1.1 Codecs

HEVC. We choose the x265 library deployed in FFMPEG software[3] as the implementation of HEVC standard. The command line for generating x265 is provided as follows,

ffmpeg -f rawvideo -video_size WxH -i input.mkv -c:v libx265 -preset veryfast -tune zerolatency -x265-params "crf=Q:keyint=GOP:verbose=1" output.mkv -y

Among them, *W,H,Q,GOP* represents the width, height, QP and GOP size, respectively. Specifically, GOP is 8 for all of the dataset.

VVC. Considering the efficiency and effectiveness, we choose VVenC[4] as the implementation of VVC standard. The command line for encoding is provided as follows,

vvencFFapp -c cfg/experimental/lowdelay_faster.cfg –InputFile input.yuv -s WxH -fr 25 –QP Q –BitstreamFile B

And the command line for decoding is provided as follows:

vvdecapp -b B -o output.yuv

Among them, *W,H,Q,B* represent the width, height, QP and bitstream of the to-be-coded videos. Note that the configuration of GOP size is set in the cfg file.

1.2 VOS Models

AOT As all of the system is conducted by PyTorch, the chosen version[5] of AOT is also implemented by PyTorch. Specifically, the model is **R50-AOTL**

[3] https://github.com/FFmpeg/FFmpeg
[4] https://github.com/fraunhoferhhi/vvenc
[5] https://github.com/yoxu515/aot-benchmark

STCN We use the official released version[6] of STCN as the implementation.

1.3 VOS Evaluation

DAVIS For DAVIS dataset, we use the valid dataset to verify the performance of the proposed framework. Thus we utilize the official released tools[7] to evaluate the performance.

YouTube-VOS As the source code of the evaluation of YouTube-VOS dataset is not provided, we evaluate the performance of YouTube-VOS on the competition server[8].

2 Complexity Analysis

2.1 Experimental results about complexity

We conduct an additional experiment to compare the runtime complexity between the baseline and our proposed framework. As the traditional codecs are not neural network based methods, it is difficult to calculate the flops to evaluate the complexity. As an alternative, we further provide the runtime analysis to measure the complexity of the whole model. The experiment is employed on the DAVIS 2017 dataset. The simulation environment is based on Ubuntu 18.04 with one NVIDIA 3080ti graphic card. The experimental results are shown in Table 1, in which t_c, t_v, t_s represent the runtime of the codecs, VOS model, and the whole framework, respectively. Table 1 presents that our framework still outperforms x265 when the runtime is similar. Note that we choose a faster VOS model to align the runtime. The experimental results also show that the state-of-art codec VVEnc (an implementation of the Volatile Video Coding (VVC) standard.) is much slower.

Table 1. Runtime Analysis on DAVIS 2017 Dataset

Codec	**VOS**	t_c	t_v	$t_s = t_c + t_s$	**Bitrate**↓	$(\mathcal{J}\&\mathcal{F})_m$↑
x265	STCN	499.5s	74.5s	**574.0s**	0.0209	0.6947
x265	AOT	499.5s	116.0s	**615.5s**	0.0209	0.7386
Ours	STCN	535.2s	74.4s	**609.6s**	0.0158	0.8265
Ours	AOT	531.7s	116.0s	**647.7s**	0.0178	0.8354
VVEnc	STCN	2475.9s	74.2s	**2550.1s**	0.0179	0.7440

[6] https://github.com/hkchengrex/STCN
[7] https://github.com/davisvideochallenge/davis2017-evaluation
[8] https://competitions.codalab.org/competitions/20127#results

2.2 Experimental results about decoding complexity

We conduct an additional experimental on DAVIS 2017 dataset to compare the runtime complexity of the encoder and the decoder. The experimental results in shown in Table 2, in which t_e, t_d, t denote the runtime on the encoder side, decoder side, and the whole framework, respectively. Table 2 presents that our proposed method is much faster than traditional codecs because the VOS model is employed on the encoder side.

Table 2. Encoder/Decoder Complexity Analysis on DAVIS 2017 Dataset

Codec	**VOS**	t_e	t_d	$t = t_e + t_d$	**Bitrate**↓	$(\mathcal{J}\&\mathcal{F})_m$↑
x265	STCN	489.7s	**84.3s**	574.0s	0.0209	0.6947
Ours	STCN	575.9s	**33.7s**	609.6s	0.0158	0.8265
x265	AOT	489.7s	**125.8s**	615.5s	0.0209	0.7386
Ours	AOT	614.2s	**33.5s**	647.7s	0.0178	0.8354

3 Supplemental Experimental Results

3.1 DAVIS 2017

In the submitted paper, we provide the J_m and F_m of the proposed framework on DAVIS 2017. In this subsection the {Recall↑, Decay↓}x{$\mathcal{J}$,$\mathcal{F}$} for DAVIS 2017 are provided.

Table 3. Supplemental Experimental Results on DAVIS 2017 dataset.

VOS Model	**Method**	**Bitrate**↓	$\mathcal{J}_r$↑	$\mathcal{J}_d$↓	$\mathcal{F}_r$ ↑	$\mathcal{F}_d$ ↓
AOT	Original	-	0.9129	0.0428	0.9430	0.0617
	x265(baseline)	0.0209	0.7987	0.0926	0.8168	0.1483
	x265+Ours	0.0178	0.9051	0.0398	0.9420	0.0566
STCN	Original	-	0.9142	0.0603	0.9458	0.0861
	x265(baseline)	0.0209	0.7923	0.1470	0.8204	0.2123
	x265+Ours	0.0158	0.8820	0.0427	0.9201	0.0727

3.2 More Samples

In this subsection, we provide more visualizations about the experimental results in Fig. 1 to Fig. 3. The first column indicates the original video sequences and the ground truth annotation. The second represents the original video sequences and masks extracted by AOT model. The third column denotes the video sequence compressed by x265 and the masks extracted by AOT. And the forth column is the video sequence compressed by x265 and then enhanced by our work.

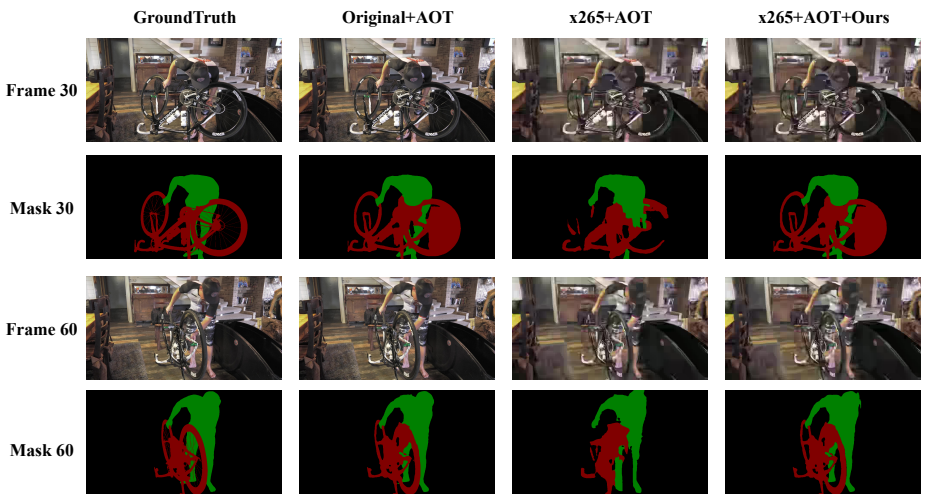

Fig. 1. Visualization of the sequence *bike-packing* in DAVIS dataset.

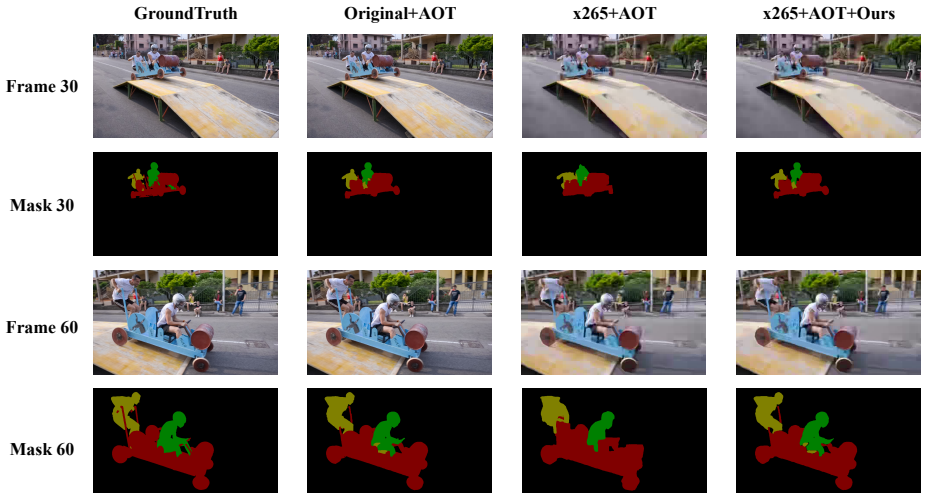

Fig. 2. Visualization of the sequence *soapbox* in DAVIS dataset.

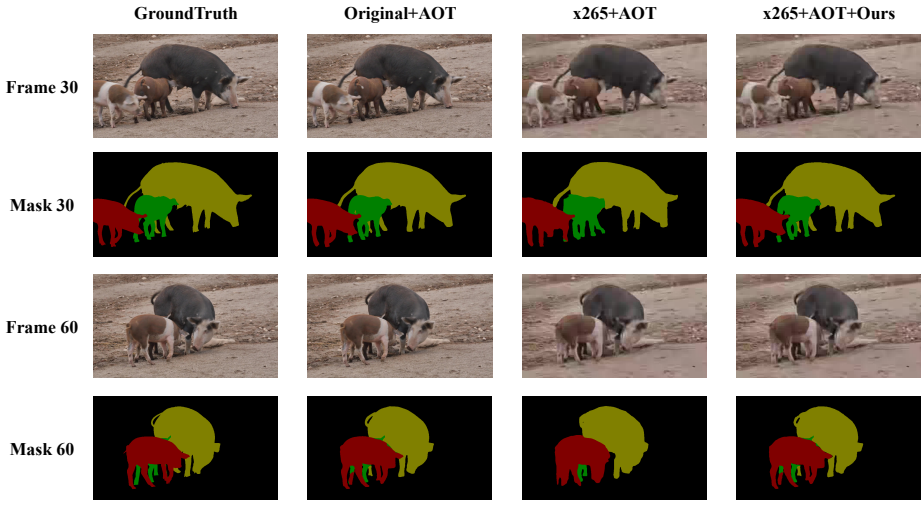

Fig. 3. Visualization of the sequence *pig* in DAVIS dataset.