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.		Supplemental Material
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, Impeq -f rawvideo -video_size WxH -i input.mkv -c:v libx265 -preset veryfast-tune zerolatency -x265-params "crf=Q:keyint=GOP:verbose=I" 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/FFmpeg/FFmpeg 4 https://github.com/Frmpeg/FFmpeg 5 https://github.com/Frmpeg/FFmpeg 5 https://github.com/Frmpeg/FFmpeg		
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, Impeq -f rawvideo -video_size WxH -i input.mkv -c:v libx265 -preset veryfast-tune zerolatency -x265-params "crf=Q:keyint=GOP:verbose=I" 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/FFmpeg/FFmpeg 4 https://github.com/Frmpeg/FFmpeg 5 https://github.com/Frmpeg/FFmpeg 5 https://github.com/Frmpeg/FFmpeg		1[0000 0001 0000 0040] 0[0000 0000 7440 0047]
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, Impeq -f rawvideo -video_size WxH -i input.mkv -c:v libx265 -preset veryfast-tune zerolatency -x265-params "crf=Q:keyint=GOP:verbose=I" 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/FFmpeg/FFmpeg 4 https://github.com/Frmpeg/FFmpeg 5 https://github.com/Frmpeg/FFmpeg 5 https://github.com/Frmpeg/FFmpeg	Zhimeng Hu	ang ¹ [0000-0001-8026-9349], Chuanmin Jia ² [0000-0002-7418-6245],
Beijing 100871, China Wangxuan Institute of Computer Technology, Peking University, Beijing 100871, China 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, Impeg-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	Shanshe W	$ang^{1[0000-0002-7665-7434]}$, and Siwei $Ma^{1[0000-0002-2731-5403]}$
Beijing 100871, China Wangxuan Institute of Computer Technology, Peking University, Beijing 100871, China 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, Impeg-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	¹ National En	gineering Research Center of Visual Technology Peking University
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, Implementation of HEVC standard. The command line for generating x265 is provided as follows, Implementation of Frawideo -video.size WxH -i input.mkv -c:v libx265 -preset veryfast -tune zerolatency -v265-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, vvenceFpapp -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 https://github.com/FFmpeg/FFmpeg https://github.com/Framphoferhhi/vvenc	radional En	9 9,7
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, fimpeg -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 https://github.com/FFmpeg/FFmpeg https://github.com/Frmpeg/FFmpeg	² Wangxuan Ii	
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, Implementation of HEVC standard. The command line for generating x265 is provided as follows, Implementation of view of the configuration of the configuration of the configuration of the configuration of the dataset. INVC. Considering the efficiency and effectiveness, we choose VVenC ⁴ as the implementation of VVC standard. The command line for encoding is provided as follows, InvencFFapp -c cfg/experimental/lowdelay-faster.cfg -InputFile input.yuv -sWxH -fr 25 -QP Q -BitstreamFile B And the command line for decoding is provided as follows: Invoked provided of the command line for decoding is provided as follows: Invoked provided of the command line for decoding is provided as follows: Invoked provided of the command line for decoding is provided as follows: Invoked provided provided provided as follows: Invoked provided		China
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, Implementation of HEVC standard. The command line for generating x265 is provided as follows, Implementation of 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 of AOT is also implemented by PyTorch. Specifically, the model is R50-AOTL The constant of the configuration of the chosen version of AOT is also implemented by PyTorch. Specifically, the model is R50-AOTL		
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, Implementation of HEVC standard. The command line for generating x265 is provided as follows, Implementation of view of the configuration of the configuration of the configuration of the configuration of the dataset. INVC. Considering the efficiency and effectiveness, we choose VVenC ⁴ as the implementation of VVC standard. The command line for encoding is provided as follows, InvencFFapp -c cfg/experimental/lowdelay-faster.cfg -InputFile input.yuv -sWxH -fr 25 -QP Q -BitstreamFile B And the command line for decoding is provided as follows: Invoked provided of the command line for decoding is provided as follows: Invoked provided of the command line for decoding is provided as follows: Invoked provided of the command line for decoding is provided as follows: Invoked provided provided provided as follows: Invoked provided		
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, In the standard of the command line for generating x265 is provided as follows, If the standard of the	1 Supplen	nental Experimental Configurations
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, In the standard of the command line for generating x265 is provided as follows, If the standard of the	In this section	the gauge ender instructions and configurations of codess and
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 https://github.com/FFmpeg/FFmpeg https://github.com/Frmpeg/FFmpeg		
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, Implementation of HEVC standard. The command line for generating x265 is provided as follows, Implementation of 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. INVC. Considering the efficiency and effectiveness, we choose VVenC ⁴ as the implementation of VVC standard. The command line for encoding is provided as follows, InvencyFapp -c cfg/experimental/lowdelay_faster.cfg -InputFile input.yuv -s InputFile inpu	VOD memous a	are provided.
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, Implementation of HEVC standard. The command line for generating x265 is provided as follows, Implementation of 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. INVC. Considering the efficiency and effectiveness, we choose VVenC ⁴ as the implementation of VVC standard. The command line for encoding is provided as follows, InvencyFapp -c cfg/experimental/lowdelay_faster.cfg -InputFile input.yuv -s InputFile inpu		
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 veryfasi-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 of AOT is also implemented by PyTorch. Specifically, the model is R50-AOTL https://github.com/FFmpeg/FFmpeg https://github.com/FFmpeg/FFmpeg https://github.com/fraunhoferhhi/vvenc	1.1 Codecs	
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 https://github.com/FFmpeg/FFmpeg https://github.com/FFmpeg/FFmpeg https://github.com/fraunhoferhhi/vvenc	HEVC. We c	hoose the x265 library deployed in FFMPEG software ³ as the
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 https://github.com/FFmpeg/FFmpeg https://github.com/fraunhoferhhi/vvenc		· - ·
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 https://github.com/FFmpeg/FFmpeg https://github.com/fraunhoferhhi/vvenc	-	
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 https://github.com/FFmpeg/FFmpeg https://github.com/fraunhoferhhi/vvenc	ffmpeg -f ra	wvideo -video_size WxH -i input.mkv -c:v libx265 -preset veryfasi
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	$-tune\ zero laten$	cy -x265-params "crf=Q:keyint=GOP:verbose=1" output.mkv -y
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	Among ther	m, W,H,Q,GOP represents the width, height, QP and GOP size
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	respectively. Sp	pecifically, GOP is 8 for all of the dataset.
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		
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	VVC. Conside	ering the efficiency and effectiveness, we choose VVenC ⁴ as the
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		· ·
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	as follows,	· ·
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		
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		
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		~ -
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		
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	_	
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		1 videos. Note that the configuration of GOP size is set in the cfg
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	піе.	
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		
is also implemented by PyTorch. Specifically, the model is R50-AOTL 3 https://github.com/FFmpeg/FFmpeg 4 https://github.com/fraunhoferhhi/vvenc	1.2 VOS Me	odels
is also implemented by PyTorch. Specifically, the model is R50-AOTL 3 https://github.com/FFmpeg/FFmpeg 4 https://github.com/fraunhoferhhi/vvenc	AOT Agall of	the system is conducted by PyTorch, the chosen version of AOT
3 https://github.com/FFmpeg/FFmpeg 4 https://github.com/fraunhoferhhi/vvenc		· · · · · · · · · · · · · · · · · · ·
4 https://github.com/fraunhoferhhi/vvenc		- · · · · · · · · · · · · · · · · · · ·
	1 // 0	, 10, 10

2 Z. Huang et al.

 ${f STCN}$ We use the official released version 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 tools⁷ 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 server⁸.

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

$\overline{\text{Codec}}$	vos	t_c	t_v	$t_s = t_c + t_s$	$\mathbf{Bitrate} \downarrow$	$(\mathcal{J}\&\mathcal{F})_m\uparrow$
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	$647.7\mathrm{s}$	0.0178	0.8354
VVEnc	STCN	2475.9s	74.2s	$2550.1\mathrm{s}$	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	$t = t_e + t_d$	$\mathbf{Bitrate} \downarrow$	$(\mathcal{J}\&\mathcal{F})_m\uparrow$
x265	STCN	489.7s	84.3s	574.0s	0.0209	0.6947
Ours	STCN	575.9s	$33.7\mathrm{s}$	609.6s	0.0158	0.8265
x265	AOT	$489.7\mathrm{s}$	$125.8 \mathrm{s}$	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 \uparrow , Decay \downarrow }x{ \mathcal{J},\mathcal{F} } for DAVIS 2017 are provided.

Table 3. Supplemental Experimental Results on DAVIS 2017 dataset.

VOS Model	Method	$\mathbf{Bitrate} \downarrow$	$\mathcal{J}_r \uparrow$	$\mathcal{J}_d \downarrow$	$\mathcal{F}_r \uparrow$	$\mathcal{F}_d\downarrow$
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 $\times 265$ and the masks extracted by AOT. And the forth column is the video sequence compressed by $\times 265$ and then enhanced by our work.

Z. Huang et al.

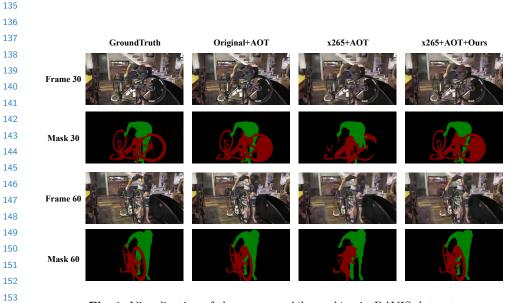


Fig. 1. Visualization of the sequence $\it bike-packing$ in DAVIS dataset.



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

CONFIDENTIAL REVIEW COPY. DO NOT DISTRIBUTE.

