Low Bitrate Compression of Video with Dynamic Backgrounds

Solomon Garber, Ryan Marcus*, Antonella DiLillo and James Storer

Brandeis University
{solomongarber,dilant,storer}@brandeis.edu

*MIT CSAIL
ryanmarcus@csail.mit.edu

Abstract: We present a technique for low bitrate two-layer sprite encoding and decoding of videos with semantically salient foregrounds and dynamic background motion, such as an outdoor sports game. By representing the background motion using static descriptors, our technique improves visual quality in the video background while preserving the fidelity of semantically important foreground regions. Our technique can target lower bitrates than may be appropriate for codecs such as AVC and HEVC.

1 Introduction

Dynamically moving background textures in video are challenging to compress. High contrast, moving video background content, such as trees blowing in the wind, require significantly more bits to encode than still backgrounds at the same visual quality or signal fidelity. For applications where a low bitrate is required, background motion can result in distracting compression artifacts. Furthermore, since standard video codecs encode the background and foreground together in a single pane, dynamic background motion will necessarily cause degradation of visual quality and signal fidelity in the semantically salient foreground video regions at a fixed bitrate. Bits saved on background encoding can be used for higher fidelity in the foreground, which may be more semantically important and visually salient.

Dynamic backgrounds may require a high bitrate to compress accurately even though they have low semantic content. In our previous work [1] we presented a lossy encoding technique for dynamic video backgrounds. This approach demonstrated that a static motion sprite representation (a sprite image, vibration modes, and a small number of per-frame global motion parameters) can be used to reconstruct videos containing dynamic oscillations without foreground object motion. By using a general model of oscillatory motion, [1] was able to achieve significantly lower bitrates than standard codecs at similar visual quality for this limited class of video.

The techniques proposed in [1] target videos containing simple oscillatory motion, and did not directly apply to more complex videos, such as those with oscillatory backgrounds but complex foregrounds (like a video of a sports game with swaying trees in the background). However, recent advances in deep learning (e.g. [2]) allow automatic pixel-level segmentation of images based on their content. These semantic masks can be used to separate foreground and background regions in traditionally challenging scenarios such as videos containing dynamic background motion.

In this work, we present an end-to-end encoder and decoder system that combines these semantic masks with the motion sprite representation of [1]. The result is a low

bitrate representation of videos containing dynamically oscillating background motion and semantically salient foreground content. Videos produced using our system have significantly better visual quality than state-of-the-art codecs like AVC and HEVC at very low bitrates, even lower than bitrates these standards intended to target.

2 Related Work

Motion Spectrum Based Video Synthesis Eulerian video motion processing [3, 4, 5, 6, 7] has allowed researchers to visualize [3, 5, 4, 6, 8], control [9], and even hear [10] various dynamic and oscillatory motions in video. Analyzing the power spectrum of the approximately oscillatory motion has been shown to reveal physical characteristics of the video objects [8, 6]. Furthermore, the work of [9] demonstrated that frequency domain slices of these motion spectra (at resonant frequencies), which had previously been used by [8] to visualize the modal shapes (amplitude and phase) of these vibration modes, could be used to reanimate a video containing such modal vibrations using arbitrary user-provided forcing functions and a simple physical simulation. Previously [11] had shown an alternate method for reanimating videos containing dynamic textures.

Content Based Video Encoding The existing MPEG-4 part 2 video standard, now almost 20 years old, offers the option of *sprite coding*, where a static video background can be represented by a single frame which is cropped at each time interval based on the angle of view of the camera. Along these lines, some work has been done on semantic segmentation for video compression, including work addressing non-oscillatory motion [12], specially encoding skin tone [13], and static texture modeling [14].

Foreground Segmentation One way to generate a sprite in the presence of foreground motion is to segment and remove the foreground. Many techniques have been developed for video foreground segmentation, often using cues from motion [15, 16]. Much recent work has addressed pixel level semantic and instance image segmentation techniques e.g. [2, 17, 18, 19], and these techniques are rapidly improving.

3 Dynamic Motion Sprites

In previous work [1], we have shown how vibration modes can be used to compress dynamically oscillating video content at low bitrates and improved visual quality. We extract modes of motion using the techniques of [8] and show how modal states can be estimated for each frame of video and used to approximately reconstruct the resonant motion dynamics of these sorts of subjects. After a brief overview of the techniques of [1], we show how these techniques can be applied to videos where the dynamic background is often or always occluded by a foreground subject exhibiting more general object motion.

In [1], a random frame from video stream containing oscillatory motion is chosen to be used as a sprite. Each frame from a section of video is then taken into the wavelet domain for phase based motion estimation. The complex phase response of these wavelets are linear in the local motion, meaning the changes in phase over time

of oriented complex valued wavelet filters can be used to estimate the motions contained in a video [5]. If the background motion is modeled as a damped oscillation, the transfer function of which is a Lorentzian distribution $\frac{s^2}{(x-\omega)^2+s^2}$ with shape and frequency parameters s and ω, then the central frequency and damping of the background video oscillations can be found by (1) processing these spatial displacements in the frequency domain and (2) finding peaks in the global amplitude response of the frame-wide motion estimates. The local motion estimates are used to generate a motion spectrum from which the modal frequencies are estimated. Once the central frequency is estimated, the slice of the frequency response of local motions at the central frequency indicates the relative amplitudes and phases of the background oscillations around that frequency at each spatial location. This frequency slice is used as a spatial basis for the dynamic motion in the background video, to be used during decoding to warp the background sprite based on global motion parameters. Once the modal frequency and shape have been determined, the background video is streamed all the way through, with motion estimates taken at each frame, filtered around the central frequency, and projected onto the modal basis using a complex inner product. The response of this inner product at each frame is stored as a global motion parameter to be used during decoding. In summary, all that is required to store the entire background video segment is three static frames and four floats of global motion information per frame.

4 Encoder

Video coding using dynamic motion sprites requires sprite generation, motion parameter estimation, and foreground object encoding. Our technique, depicted in Figure 2, uses segmentation masks generated by a pre-trained Mask R-CNN [2] model as foreground masks, to generate separate foreground and background video streams. Missing background regions are filled in for dynamic motion sprite generation, and the foreground video region is encoded using either HEVC or AVC. We also keep a low resolution video of the segmentation mask, which is spatially upscaled during decoding and used as a guide for multiplexing the two video streams together.

4.1 Semantic Foreground Subtraction

In order to generate the background sprite, motion modes, and global motion parameters, as described in [1], we first remove the foreground objects from the input video to produce a clean stream of background video for the sprite encoding. If the foreground motion is still visible in the background video stream, or if it is replaced by other motion artifacts in these regions, the oscillatory motion assumption of the motion sprite algorithm will be in error and the modes and motion parameters may produce visual artifacts reflecting these errors. Simply blacking out the masked area does not remove the appearance of motion from the video stream, and inpainting on a per-frame basis is prone to introducing motion and texture artifacts which will also corrupt the sprite generation process. If the background were static, or only gradually changing, holes in the background video could be filled directly with the background from the previous frame, as in [20]. We take a similar strategy, however since our backgrounds are moving dynamically, this approach would introduce sharp spatial

Figure 1: Sample frames from three versions of a 1 minute 50 seconds 1080p video, 23.976 fps, a total of 2638 frames. *Top*: video encoded with AVC at original quality (469,742,075 bytes, 0.687 bpp, approximately 34 Mbit/s). *Middle*: same frame encoded by AVC at low quality settings (3,544,854 bytes, .0052 bpp, approximately 258 Kbit/s). *Bottom*: same frame encoded with our method (3,508,588 bytes or .0051 bpp, approximately 255 Kbit/s). The right two columns show closeups of the visual quality achieved. Click on the still image to view the corresponding video or go to: **rm.cab/semvid1**

discontinuities in the resulting video. To mitigate this issue, we take the masked region from the previous frame, and morphologically dilate the mask as shown in Figure 3 to create a buffer region separating the current background from the previous background, and fill in this region with inpainted content.

Once the foreground has been removed from the video feed, the background stream can be processed and compressed into static sprite, motion modes, and global motion signal as in [1]. The modes are compressed using JPEG image compression (amplitude and phase are stored separately with amplitude stored in log scale) as is the background sprite. The global motion signal is stored without compression, but since they only require 4 floats per frame of video the size of this signal is minimal compared to the other components of the background representation, even without compression. JPEG quality of the sprite as well as the modes can be used to control the bitrate of the background sprite, however since the size of the background representation is

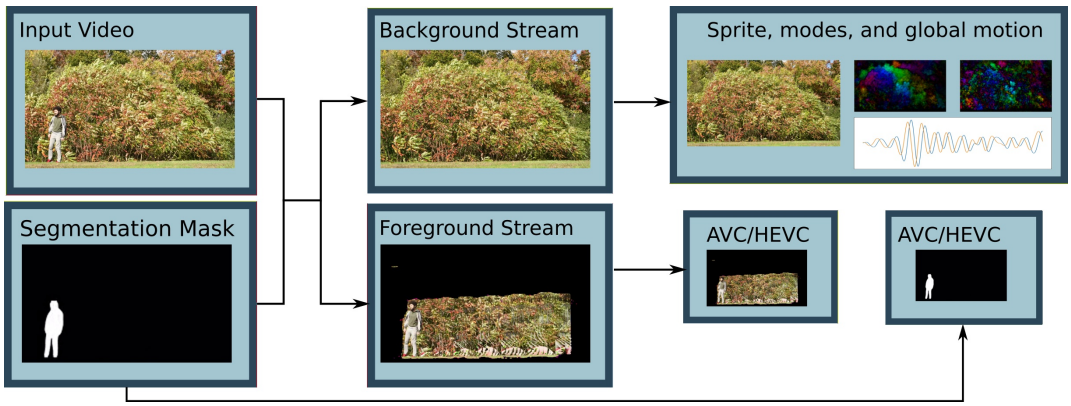

Figure 2: Our two level video encoder pipeline. Foreground regions are separated and compressed using a standard codec with image persistence. Masks are spatially downsampled and compressed with a standard codec. Background regions are filtered and reduced to a static sprite, complex valued modes of horizontal and vertical motion, and frame-wide global motion parameters. Frames are taken from the video shown in Figure 1.

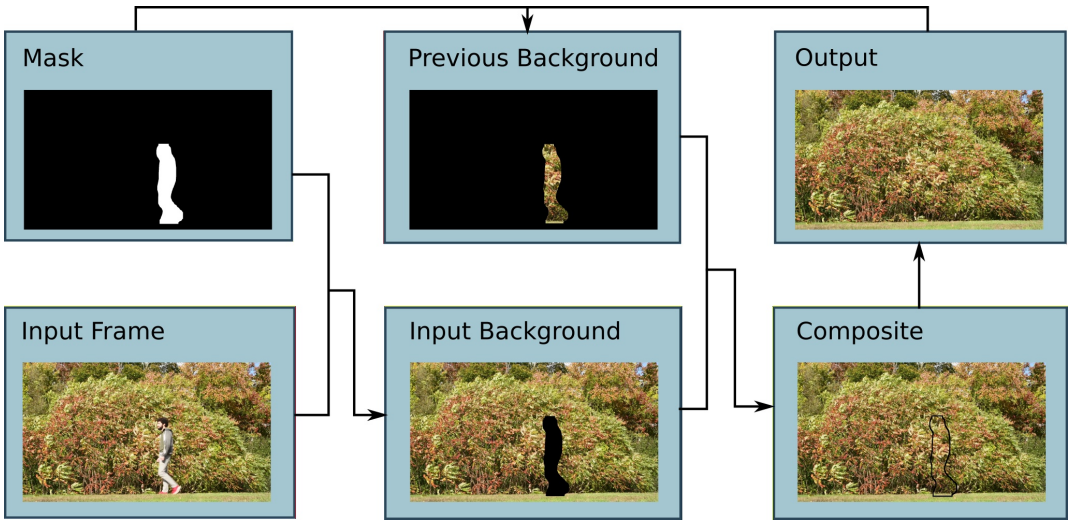

Figure 3: Our semantic segmentation based foreground subtraction feedback pipeline. Foreground regions are removed and filled in using feedback and inpainting. Frames are taken from the video shown in Figure 1

almost unaffected by the duration of the video, the sprite image quality can be much higher than the foreground video stream with minimal impact on the overall bitrate.

4.2 Foreground Object Compression

Once the foreground has been separated, it must be stored in such a way that is amenable to fast decoding and low bitrates. The foreground and mask videos are encoded separately, using AVC and HEVC in our experiments. Where possible we use HEVC, however, on some videos our technique targets lower bitrates than the HEVC tool we used (ffmpeg libx265) allows (this is the case, for example in Figure 1). In

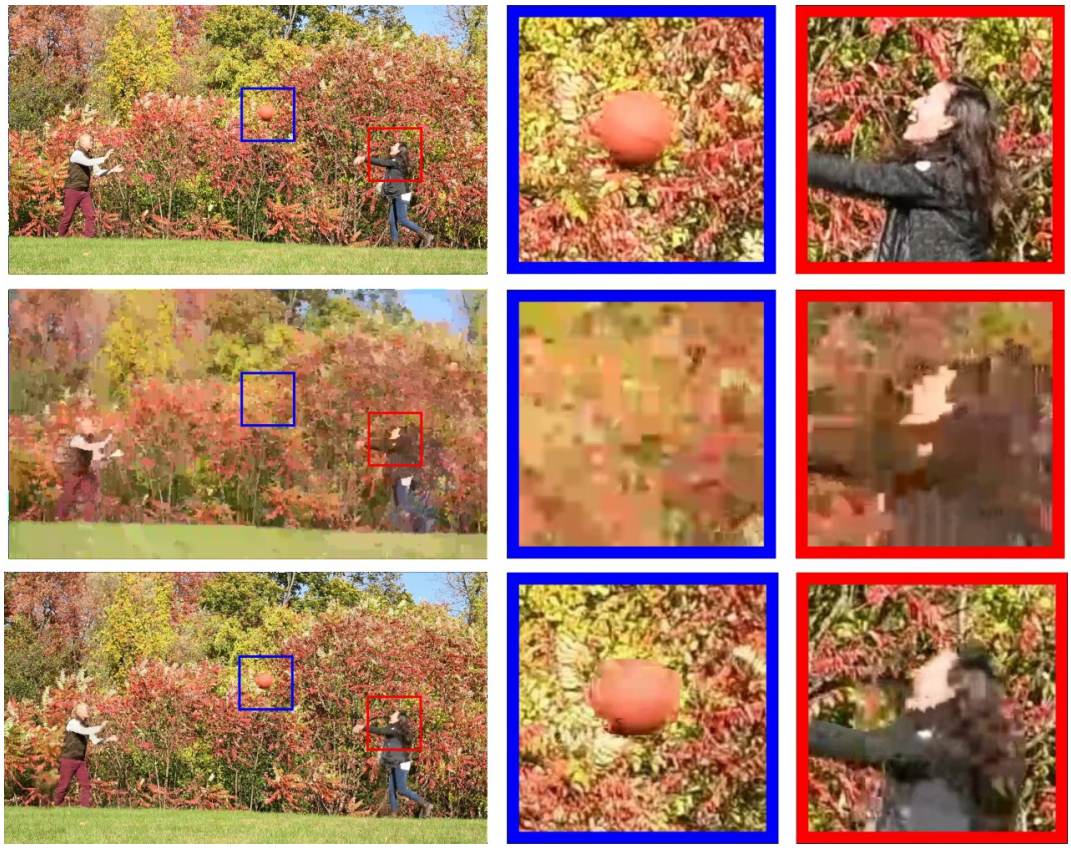

Figure 4: Sample frames from three versions of a 1 minute 48 seconds 1080p video, 59.94 fps, a total of 5868 frames. *Top*: video encoded with AVC at original quality (397,928,297 bytes, 0.26 bpp, approximately 29 Mbit/s). *Middle*: same frame encoded by AVC at low quality settings (3,438,857 bytes, .0023 bpp, approximately 255 Kbit/s). *Bottom*: same frame encoded with our method (3,436,422 bytes or .0023 bpp, approximately 255 Kbit/s). The right two columns show closeups of the visual quality achieved. Click on the still image to view the corresponding video or go to: **rm.cab/semvid4**

such instances, the AVC ffmpeg encoder, which was able to achieve lower bitrates, was employed (in these examples, for the purposes of comparison, we also used AVC to encode the foreground). The foreground regions generated by Mask R-CNN are large enough that the mask video can be stored at a significantly lower resolution than the input video, to be upsampled during decoding. For codecs that do not support video object encoding, such as the ffmpeg HEVC library libx265, storing the foreground video only with the background entirely blacked out can increase ringing artifacts in the foreground as a result of increased contrast between the foreground object and black background. Furthermore, the temporal and spatial irregularities in the masks create challenging videos for these standard codecs. To reduce the ringing artifacts in the foreground, we dilate the mask region to preserve contrast at the object boundaries. To mitigate the temporal artifacts of masked regions appearing and disappearing, we make a foreground stream where nothing is ever deleted until

Figure 5: Sample still frames taken from mask video before image persistence is added (top) and their corresponding frames with image persistence. (bottom) Frames are taken from the foreground of the video shown in Figure 4

it is written over, creating a foreground video with image persistence as in Figure 5.

5 Decoder

The final encoded video contains a static background sprite, vertical and horizontal complex motion modes, a time domain global motion signal, a foreground video and a mask video. For each frame of input video, the motion modes are scaled by their respective complex motion parameter and used as a dense flow field to warp the background sprite as in [1]. The corresponding foreground and mask frames are then decoded, and the mask frame is upscaled to the full video resolution using bicubic interpolation. A threshold is applied to the mask followed by morphological operations to clean up discontinuities due to compression artifacts. The mask is then used to overlay the foreground video stream onto the dynamically warped background.

6 Case Study: Outdoor Soccer Video

A natural application for our system is low bandwidth streaming of outdoor sports events such as a soccer game. We compare our method to HEVC (ffmpeg libx265) on a 1080p video clip of a soccer game at Brandeis University where players are moving in the foreground and trees are blowing in the background. The video clip is 2 minutes and 40 seconds in duration, shot at 23.976 frames per second, a total of 3714 frames. In the first experiment the field of view is fixed, covering only the eastern half of the field. In the second experiment we simulate a panning camera capable of following the play, by dynamically cropping a 4k video that covers the whole field.

Stable Camera The clip when encoded with our HEVC tool at its default good quality setting was 62,688,079 bytes (0.065 bpp, approximately 3 Mbit/s). With our HEVC tool at its lowest quality setting, the clip was 2,040,633 bytes (0.0021 bits per pixel, approximately 102 Kbit/s). Our encoder produced a clip of 2,032,943 bytes (0.0021 bits per pixel, 102 Kbit/s). Of those bytes, only 410,033 bytes were

Figure 6: Still frame from the soccer video (left), the video compressed by HEVC (middle), and our method (right). Click on the still image to view the corresponding video or go to **rm.cab/semvid6**

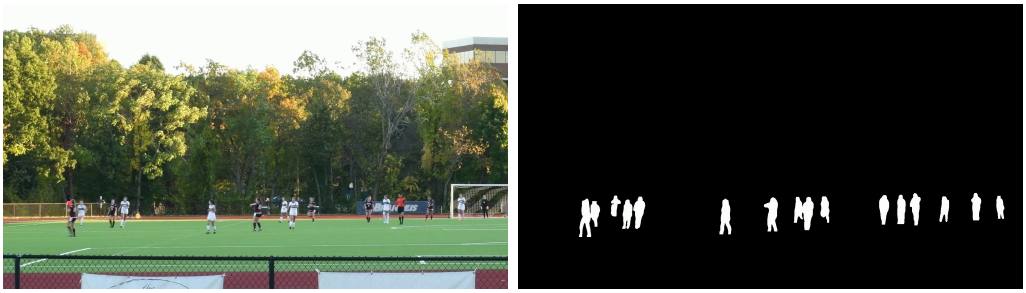

Figure 7: A frame taken from our static soccer video (left) and the corresponding foreground mask generated by Mask R-CNN (right)

used for the background, leaving more bytes for encoding only the foreground using HEVC. As described in section 3 and [1], adding successive frames to this background representation requires only 2 complex numbers per frame. For example, doubling the length of our example video (from 3714 frames to 7428 frames), and assuming 8 bytes per complex number, the additional space required for the background would result in only 59,424 bytes, allowing for even more bits to represent the foreground at the same bitrate. The results of this experiment are shown in Figure 6. As can be seen from the video, the masks produced by Mask R-CNN are not perfect, and sometimes miss the ball or partially occluded players. However the results are good enough to facilitate a higher overall visual quality than standard codecs at the same bitrate despite some irregularities.

Simulated Camera Pan In practice, sports games are often video taped with a panning camera that is following the action. However, a different approach, which is feasible with current technology, is to dynamically crop a high resolution video (e.g. 4k or higher) that covers the whole field. While it may be possible to recover dynamic motion sprites under camera panning conditions, it is certainly simpler with a still camera to remove the need of estimating camera position as well as eschewing motion blur introduced by analog camera motion. For example, [21] describes an automatic system for dynamically cropping high resolution hockey footage around the action.

Along these lines, a 4k resolution video was dynamically cropped to simulate a camera operator panning across the field, as shown in Figure 8. Simulating panning after the fact allowed us to create a 4k motion sprite the size of the whole field. For

this example, the mask was cropped to simulate the viewing angle of the panning, but the foreground video with image persistence was stored at the full resolution, with only visible regions changing at each frame. The sprite, motion modes, and global motion signal were generated from the full resolution video. The cropping parameters were stored along with the global motion parameters and used during decoding.

Figure 8: Three sample frames from our compressed simulated camera pan video. On the left is shown the position of the frame on the warped background sprite, and on the right is our decoded frame. Click on the still image to view the corresponding video or go to **rm.cab/semvid8** where a comparison with HEVC can also be seen.

The clip when encoded with our HEVC tool at its default good quality setting was 63,338,204 bytes (0.066 bits per pixel, approximately 3 Mbit/s). With our HEVC tool at its lowest quality setting, the clip was 2,557,625 bytes (0.0027 bits per pixel, aprroximately 128 Kbit/s). We then set our encoder to approximately match the same rate (2,557,435 bytes, 0.0027 bits per pixel, or approximately 128 Kbit/s). Our encoder compressed the background of this video to only 530,689 bytes, leaving more bits for encoding only the foreground using HEVC.

7 Conclusion

We have presented a technique that can be used for very low bitrate video compression where visual quality may be improved over encoding with AVC or HEVC at

comparably low rates. The method incorporates our previous work for low cost static representations of dynamic backgrounds allowing bits that are no longer needed to produce a visually acceptable background to be allocated to the semantically more important foreground material. Acquisition of the video at high resolution allows for the construction of larger dynamic sprites that can support a panning field of view. We expect that continued improvements in machine learning algorithms for automatic segmentation will not only improve the quality of our system at low bitrates but also allow the possibility of employing it at higher quality. In addition, our system would benefit from improved compression of foreground video objects.

References

[1] S. Garber et al. "Compact Representations of Dynamic Video Background Using Motion Sprites". In: *2019 Data Compression Conference (DCC)*. IEEE. 2019.

[2] K. He et al. "Mask R-CNN". *ICCV* (2017), pp. 2980–2988.

[3] H.-Y. Wu et al. "Eulerian video magnification for revealing subtle changes in the world". *TOG* 31 (2012), 65:1–65:8.

[4] N. Wadhwa et al. "Phase-based video motion processing". *TOG* 32.4 (2013), p. 80.

[5] N. Wadhwa et al. "Riesz pyramids for fast phase-based video magnification". *ICCP* (2014).

[6] J. G. Chen et al. "Video camera–based vibration measurement for civil infrastructure applications". *Journal of Infrastructure Systems* 23.3 (2016).

[7] E. Prashnani et al. "A Phase-Based Approach for Animating Images Using Video Examples". *Computer Graphics Forum* 36.6 (2017).

[8] A. Davis et al. "Visual vibrometry: Estimating material properties from small motion in video". *CVPR* (2015), pp. 5335–5343.

[9] A. Davis, J. G. Chen, and F. Durand. "Image-space modal bases for plausible manipulation of objects in video". *TOG* 34.6 (2015), p. 239.

[10] A. Davis et al. "The visual microphone: passive recovery of sound from video". *TOG* 33.4 (2014), p. 79.

[11] G. Doretto et al. "Dynamic textures". *IJCV* 51.2 (2003), pp. 91–109.

[12] A. Smolic, T. Sikora, and J.-R. Ohm. "Long-term global motion estimation and its application for sprite coding, content description, and segmentation". *IEEE Transactions on Circuits and Systems for Video Technology* 9.8 (1999), pp. 1227–1242.

[13] U. Khan, M. Cheema, and N. Sheikh. "Adaptive video encoding based on skin tone region detection". In: *ISCON*. Vol. 1. IEEE. 2002, pp. 129–134.

[14] A. Dumitras and B. G. Haskell. "An encoder-decoder texture replacement method with application to content-based movie coding". *IEEE Transactions on Circuits and systems for Video Technology* 14.6 (2004), pp. 825–840.

[15] M. Kunter, J. Kim, and T. Sikora. "Super-resolution mosaicing using embedded hybrid recursive folow-based segmentation". In: *ICSPIC*. IEEE. 2005, pp. 1297–1301.

[16] A. Krutz et al. "Motion-based object segmentation using sprites and anisotropic diffusion". In: *Eighth International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS'07)*. IEEE. 2007, pp. 35–35.

[17] O. Ronneberger, P. Fischer, and T. Brox. "U-net: Convolutional networks for biomedical image segmentation". In: *MICCAI*. Springer. 2015, pp. 234–241.

[18] R. Hu et al. "Learning to segment every thing". In: *CVPR*. 2018, pp. 4233–4241.

[19] A. Kirillov et al. "Panoptic segmentation". In: *CVPR*. 2019, pp. 9404–9413.

[20] S. Garber et al. "Visual Lecture Summary Using Intensity Correlation Coefficient". In: *IMVIP*. 2017, pp. 68–75.

[21] H. Pidaparthy and J. Elder. "Keep Your Eye on the Puck: Automatic Hockey Videography". In: *WACV*. IEEE. 2019, pp. 1636–1644.