EVENT-DRIVEN VIDEO CODING FOR OUTDOOR WIRELESS MONITORING CAMERAS

Zichong Chen, Guillermo Barrenetxea and Martin Vetterli

{zichong.chen, guillermo.barrenetxea, martin.vetterli}@epfl.ch
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne (EPFL), Lausanne CH-1015, Switzerland

ABSTRACT

Reducing communication cost is crucial for outdoor wireless monitoring cameras which are constrained by limited energy budgets. From event detection point of view, traditional video coding schemes such as H.264 are inefficient as they ignore the "meaning" of video content and thus waste many bits to convey irrelevant information. To take advantage of the powerful computing resource on cameras, we propose a novel event-driven video coding scheme. Unlike previous approach that attempts to find anomalous image frame with potential events, we propose to detect salient regions in each image and transmit the image fragments marked with saliency to the receiver. This scheme rarely drops an event as it transmits all image fragments with potential events, and also requires no training procedure. The experimental results show that it performs substantially better than conventional video coding schemes for outdoor monitoring task.

Index Terms— video coding, event-driven, saliency map, wireless camera, outdoor monitoring

1. INTRODUCTION

Thanks to the availability of low-cost image sensor chips, wireless monitoring cameras begin to emerge in various outdoor monitoring applications [1] [2]. In these tasks, one or more static monitoring cameras are deployed to detect some unknown events, such as natural hazards or unexpected intrusions. To detect such random events, cameras are usually programmed to capture a scenery of interest periodically (1-200 images/hour in our case), and transfer the image sequences back to a base station (BS). A user at the BS can inquiry the recorded images to determine if there is any particular events occurred.

Under the severe energy constraints of wireless cameras, i.e., nodes operating on batteries and a solar panel, we need to minimize the transmitted data size because the radio transceiver is often the largest energy consumer. A typical solution is to apply a video coding scheme such as H.264 [3] to compress the raw image sequence. However, in practice, this strategy is inefficient as the conventional video coding ignores the "meaning" of video content, and therefore it wastes many bits to convey irrelevant information such as cloud movements, changes in light condition, shadows, etc..

To take advantage of the powerful computing power, we can further analysis the raw image sequence on the camera before it attempts to transmit anything, that is, to determine which frame does contain an event and only to transmit such event-frames. This is related to the research on anomaly detection [4], which utilizes pattern recognition techniques to find patterns in data that do not conform to regular behavior. Similar to all recognition algorithms, there are two main drawbacks if we adopt this idea for video transmission: i) It will miss some event-frames because recognition does not work 100% accurately. ii) It requires to collect training data for supervised learning, which is time-consuming as it varies from task to task.

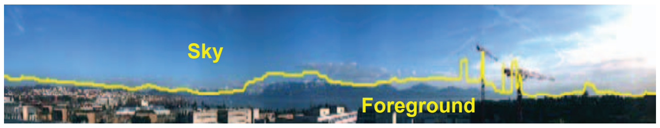


Fig. 1. Algorithm 1 applied to an image sequence of a webcam: the sky and the foreground buildings/mountains are well separated by the yellow line.

In this paper, we investigate the idea of event-driven video transmission and propose a novel video coding scheme. To avoid the risk of missing an event-frame, our scheme does not determine which frame contains an event, instead, it first analyze important changes between subsequent frames. The located salient pixels usually include the events of interest, which is denoted as *saliency* in this paper. As this procedure removes the "background", event-less regions of the image, it is sufficient to transmit the image fragments marked with saliency. In this way, as our proposed saliency detection algorithm can filter out most of the irrelevant changes in image sequence, this scheme is able to transmit substantially less bits while all image fragments with potential events are delivered to the receiver. It is designed to achieve the following targets:

- Reduces the communication rate as opposed to conventional video coding schemes such as H.264, while no potential event is missed at the receiver.
- Requires no supervised learning procedure.
- The algorithm has low complexity which can be easily implemented on an embedded platform.

The remainder of this paper is organized as follows: To avoid the disturbance of cloud movements, in Section 2, we propose a sky extraction algorithm to generate a region of interest (non-sky part). Then, in Section 3, we explain the saliency detection algorithm that locates salient regions in non-sky part of each image frame. Section 4 depicts the entire event-driven video coding system. Finally, in Section 5 we evaluate the proposed coding scheme, and the results show that our event-driven video coding scheme performs substantially better than the conventional video coding method.

2. SKY EXTRACTION

In outdoor monitoring tasks, the sky is usually included in images. Clouds in the sky move randomly and thus are difficult to distinguish from other salient regions. To filter out the entire sky part because no events of interest are expected to happen in it, we define a sky mask and apply saliency detection only to the non-sky part. To auto-

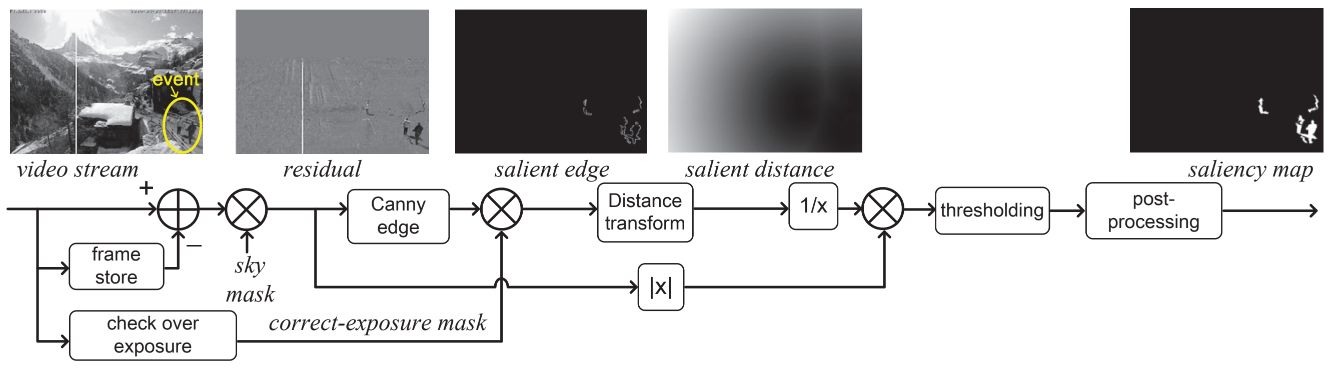


Fig. 2. Flow chart of the proposed saliency detection algorithm. The italic words represent the image signals generated throughout the algorithm, and the corresponding sample images are depicted above each signal name.

Algorithm 1 Pseudocode of sky extraction algorithm

```
1:  initialize WEIGHT: scaled in a squared manner w.r.t height
2:  LAST=1st image frame, RESIDUAL=0, COUNT=0
3:  threshold THRE= 40, refreshing period PERIOD=20
4:  while a new image CURRENT is loaded do
5:      RESIDUAL = RESIDUAL + abs(CURRENT-LAST)$\times$WEIGHT
6:      MASK = normalize(RESIDUAL) $>$ THRE
7:      image erode and dilate applied to MASK to remove small fragments
8:      if COUNT%PERIOD==1 then
9:          RESIDUAL = RESIDUAL$\times$MASK
10:     end if
11:     COUNT++
12:     LAST=CURRENT
13: end while
14: output MASK as the sky part
```

matically detect the sky part in images, we propose a sky extraction algorithm. It is based on two observations. First, clouds in the sky are dynamic, while camera and the non-sky part are comparatively static. Secondly, the sky is at the top of the images, and the non-sky part is at the bottom.

The details of sky extraction is described in Algorithm 1. The main idea is to calculate the accumulative residual image from successive image frames, and then to apply morphology operations and thresholding in order to obtain a mask of the sky part. As the sky is more dynamic compared to the foreground, it has a higher residual value. We can discriminate the sky from the foreground through the residual value. Some reflective facets of buildings may vary greatly as light condition changes. To distinguish this with the sky part, we explore the second observation, i.e., weighting the residual map according to the vertical height in a squared manner. Fig. 1 shows an example of the sky extraction result. It can be seen that most of the foreground buildings and mountains are well separated from the sky.

3. SALIENCY DETECTION

The core concept of our event-driven video coding scheme is to transmit the image fragments that are likely to have a potential event. Therefore, before the camera transmits anything, it should analysis the image sequence and locate those salient regions in each frame (denoted as *saliency map*). Naturally, the residual of successive image frames provides saliency information because an event will be visually distinguishable. We can apply thresholding to the residual to obtain a saliency map. However, due to the changes in light condition, the correct threshold value can vary within the same image. To avoid this problem, we propose to first use the edge of residual to detect the shape of salient regions. As the edge detection algorithm [5] is based on the local gradient value, it is more robust with respect to the intensity changes.

Figure 2 illustrates the routine of our saliency detection algorithm. The raw video stream is fed into the algorithm from the left side. The residual image is calculated by storing the most recent frame. After applying the sky mask to the residual image, the salient edges are computed by using a Canny edge detector [5]. To eliminate the interference of the sunlight, the raw image is checked, and the over-exposure pixels in the salient edges are cleaned up. In order to expand the shape of salient regions into solid salient regions, we first compute the distance transformation [6] of the salient edges, then combine the distances to salient edges and the residual values to represent the saliency strength of each pixel. By using thresholding, the final saliency map is generated. Post-processing techniques such as binary morphology operators can be used to refine the results. The detailed steps are explained in Algorithm 2. From the sample results provided in Figure 2, we can see that most of the cloud movements and the changes caused by light condition are filtered out, and the event of interest is correctly located by the saliency map.

Algorithm 2 Pseudocode of saliency detection algorithm (the italic words represent the image signals as in Figure 2.)

```
1: The raw video stream is buffered to generate a *residual* image (current frame minus the stored frame).
2: The *residual* image is masked so that the entire sky part is reset to zero. The *sky mask* is provided by Algorithm 1.
3: The *salient edge* is calculated from the *residual* image using a Canny edge detector [5], and then masked so that the over-exposure part is reset to zero.
4: The *salient distance* is calculated as the distance transformation [6] of the *salient edge*.
5: The absolute value of the *residual* is divided by the *salient distance* (per-pixel calculation). Thresholding is applied to the result to get an initial *saliency map*.
6: The *saliency map* is refined with some post-processing, such as removing small fragments, filling holes, etc..
```

Algorithm 3 Pseudocode of event-driven video encoding

```
1: GOP: $N$, one-byte header: *id*, counter: $k$ is initialized to 1
2: dimension of image after divided into blocks: $bh \times bw$
3: position array: $pos[bh][bw]$
4: decoding buffer: *mem* is initialized to all-zero
5: while a new image frame *cur* is captured do
6:     if ($k$ mod $N$) == 1 then
7:         compress *cur* using JPEG encoder, *id*=1
8:     else
9:         reset array *pos* to all-zero
10:        run Algorithm 2, get a *saliency map*, and divide it into
           $bh \times bw$ blocks
11:        locate the blocks marked with saliency, and set corre-
           sponding element in array *pos* to 1
12:        if sum(*pos*)/($bh \times bw$) $> 10\%$ then
13:            compress *cur*$-$*mem* using JPEG encoder, *id*=2
14:        else
15:            extract *event blocks* from *cur* according to array *pos*
16:            concatenate *event blocks* into an image strip, and com-
               press it using JPEG encoder, *id*=3
17:        end if
18:    end if
19:    transmit the JPEG codeword to the receiver
20:    if *id*==3 then
21:        compress array *pos* using a data compression library such
           as zlib, and transmit the codeword to the receiver
22:    else
23:        transmit one-byte *id* to the receiver
24:    end if
25:    update *mem* by reconstruct *cur* using transmitted codewords
26: end while
```

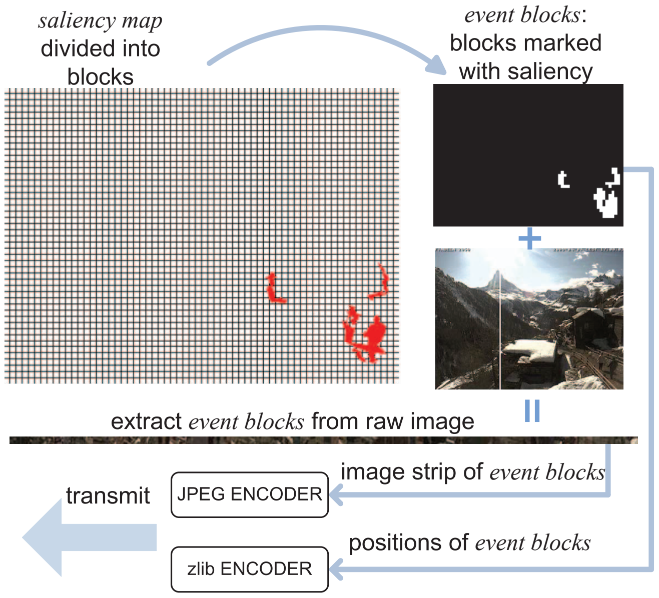
4. CODING SYSTEM

After the saliency map is generated for each image frame, we now explain how to use it in our proposed event-driven video coding system. First, similar to all video coding scheme, we define the group of pictures (GOP): an independent key-frame is compressed and transmitted every N frames. We can use a conventional image coding scheme for such compression. Without loss of generality, we use JPEG codec in this paper. The coding procedure is shown in Figure 3: For each frame except key-frame, we divide the generated saliency map into blocks of 8×8 pixels[1], and locate the blocks marked with saliency (denoted as the *event blocks*). If the percentage of the *event blocks* is more than 10%, then we encode the current frame using DPCM [7]. Otherwise, all the *event blocks* are concatenated into an image strip for compression and transmission. In the later case, we also need to send the position array of those *event blocks* so that the reconstruction is possible. In practice, a standard data compression scheme such as zlib[2] can be used to reduce the size of this position array before transmission. The detailed algorithm for the encoding procedure is shown in Algorithm 3.

At the receiver side, the decoding procedure is straightforward by following the encoding procedure (omitted for brevity). It is worth to mention that when pasting the *event blocks* onto the decoding buffer, it is actually an image blending problem. To reduce the block effect, we can first create a blending mask using the position array of *event blocks*, and then apply blending algorithms such as pyramid blending [8], Poisson image editing [9] to achieve seamless stitching.

[1]This is to meet the block size of JPEG standard.

[2]See `http://zlib.net/`.



saliency map divided into blocks
event blocks: blocks marked with saliency
+
=
extract *event blocks* from raw image
image strip of *event blocks*
transmit
JPEG ENCODER
positions of *event blocks*
zlib ENCODER

Fig. 3. Illustration of event-driven video encoding procedure. *event blocks* are extracted according to the *saliency map*, and concatenated into an image strip for transmission.

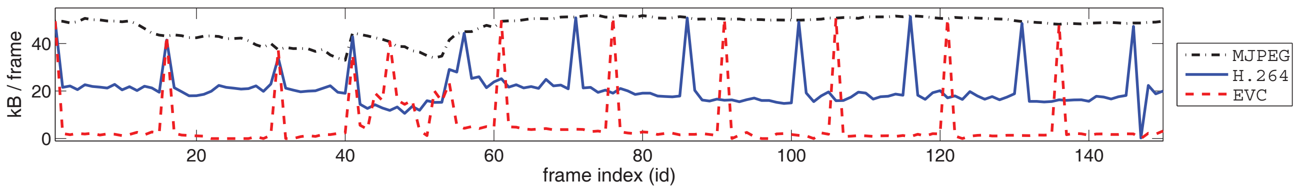
5. EXPERIMENTS

In this section, we evaluate the compression ratio of the proposed event-driven video coding (EVC). To simulate the algorithm, we downloaded a sequence of images captured by an outdoor webcam near Zermatt, Switzerland. We use 150 images which are recorded at every 5 minutes during daytime. To make a comparison, we evaluate three video coding schemes, namely, Motion JPEG, H.264, and EVC. In the experiment, the compression quality of Motion JPEG, H.264 and the transmitted image fragments of EVC are all set as equal. The GOP of EVC and H.264 are both set to 15. All the raw images and the decoded images of EVC can be accessed at `http://dl.dropbox.com/u/7084673/papers/eventcoder_data.zip`.

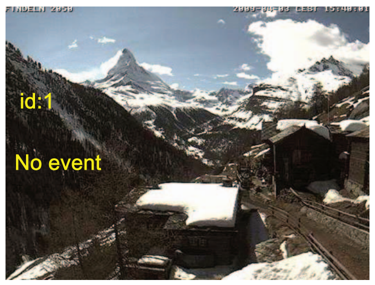
Figure 4a shows encoded frame sizes of the three schemes respectively. It can be seen that EVC performs substantially better than the other video coding schemes. The average transmitted data of EVC is reduced by 70% as compared to H.264. We also show some sample reconstructions of EVC: Figure 4b shows a decoded image with no event for reference purpose. Figure 4c shows two decoded images with events. The positions of the transmitted image fragments are also illustrated, which clearly show the detected events.

6. CONCLUSIONS

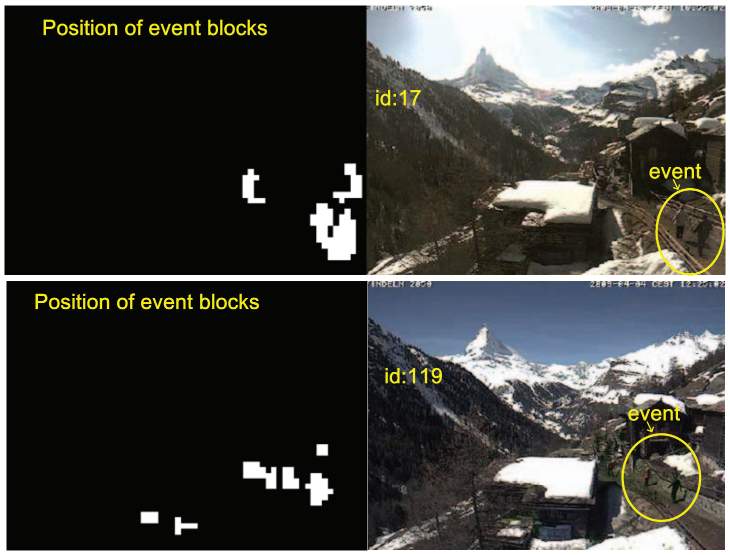
We propose an event-driven video coding system for outdoor monitoring tasks. To avoid the disturbance from the cloud movements in sky, we first propose a sky extraction algorithm to generate a sky mask for the rest of the system. By detecting the salient regions in the non-sky part of each image frame, the camera can locate the



(a)



(b)



(c)

Fig. 4. Experimental results of Motion JPEG (MJPEG), H.264, and EVC. (a) Encoded frame sizes of the three schemes respectively. Each spike in the curve represents a key-frame or DPCM frame. (b) A reference image (id=1) with no event. (c) Two EVC decoded images (id=17, 119) with detected events, and the positions of the transmitted event blocks.

image fragments with potential events. As our saliency detection algorithm can filter out most of the irrelevant changes in image sequence, this scheme is able to transmit substantially less bits while potential events are delivered to the receiver. We evaluate the system performance and the results show that the proposed system performs substantially better than conventional video coding schemes. For future work, we are planning to deploy this system in the Swiss Alps for monitoring natural hazards such as avalanche. Initial experimental results on a real wireless camera prototype have shown good energy saving and similar computation complexity as compared to the H.264 compression scheme.

7. ACKNOWLEDGEMENTS

This research was supported by the National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS, http://www.mics.org), and the ERC Advanced Investigators Grant of European Union.

8. REFERENCES

[1] John Hicks, Jeongyeup Paek, Sharon Coe, Ramesh Govindan, and Deborah Estrin, “An easily deployable wireless imaging system,” in *ImageSense’08: Proceedings of the Workshop on Applications, Systems, and Algorithms for Image Sensing*, 2008.

[2] Zichong Chen, Guillermo Barrenetxea, and Martin Vetterli, “Share Risk and Energy: Sampling and Communication Strategies for Multi-Camera Wireless Monitoring Networks,” in *Proceedings of the 31st Annual IEEE International Conference on Computer Communications (INFOCOM 2012)*, 2012.

[3] J. Ostermann, J. Bormans, P. List, D. Marpe, M. Narroschke, F. Pereira, T. Stockhammer, and T. Wedi, “Video coding with H. 264/AVC: tools, performance, and complexity,” *IEEE Circuits and Systems magazine*, vol. 4, no. 1, pp. 7–28, 2005.

[4] Tao Xiang and Shaogang Gong, “Video behavior profiling for anomaly detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 5, pp. 893 –908, may 2008.

[5] John Canny, “A computational approach to edge detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679 –698, nov. 1986.

[6] G. Borgefors, “Distance transformations in arbitrary dimensions,” *Computer vision, graphics, and image processing*, vol. 27, no. 3, pp. 321–345, 1984.

[7] A.N. Netravali and B.G. Haskell, *Digital pictures: representation, compression, and standards*, Plenum Publishing Corporation, 1995.

[8] P.J. Burt and E.H. Adelson, “A multiresolution spline with application to image mosaics,” *ACM Transactions on Graphics (TOG)*, vol. 2, no. 4, pp. 217–236, 1983.

[9] P. Pérez, M. Gangnet, and A. Blake, “Poisson image editing,” in *ACM Transactions on Graphics (TOG)*. ACM, 2003, vol. 22, pp. 313–318.