IMPROVING ERROR RESILIENCE OF SCALABLE H.264 (SVC) VIA DRIFT CONTROL

Wai-tian Tan, Andrew Patti

Multimedia Communications and Networking Lab
Hewlett Packard Laboratories, Palo Alto

ABSTRACT

Common error concealment schemes mitigate errors for frames in which losses occur only, even though errors propagate to future frames. Drift control is generally challenging due to lack of reliable basis for determining what needs to be corrected and how. In this paper, we show that for scalable or multi-layer video, an available base-layer can serve as such basis to allow continous error drift checking and correction of higher layers even when the base-layer is of much lower spatial resolution. The associated algorithm is low-complexity, incurs no additional bit cost, and experiments using SVC reference software show PSNR improvement of up to 5 dB over concealment methods without drift control.

Index Terms— Scalable video, Error resilience, Error concealment, Drift control

1. INTRODUCTION

The use of scalable video compression is an attractive solution to many video communication tasks involving multiple clients of heterogeneous bandwidth and display sizes, where each client receives a number of layers commensurate with its available resources. The latest and most promising scalable compression standard is H.264 SVC, for which an overview can be found in [1]. Nevertheless, scalable compression itself does not necessarily provide a more error resilient bit-stream. Instead, the separation of compressed data into multiple layers makes it easier to strategically offer more protection to the more important lower layers to possibly achieve better system error resilience. Differential protection can be realized by available network QoS mechanisms, or emulated by differential application of intra-coding or error control codes.

For a client that receives multiple layers, presumably with more losses for higher layers, one strategy is to only use the lower layers with insignificant losses, and discard other layers. Obviously, this avoids catastrophic image breakup commonly associated with high packet loss rates. Nevertheless, discarding of correctly received data is clearly suboptimal, especially when the discarded layers are of substantial bit-rate.

Along these lines, an overview of H.264 SVC error resilience strategies, including inter-layer techniques, as well as those requiring multi-loop decoding, can be found in [2]. One approach, referred to as *BLSkip*, is to employ *inter-layer* error concealment where missing information in a desired layer is estimated from lower layers. Under *BLSkip*, a missing block is concealed by the texture of the corresponding base-layer block if it is intra-coded. Otherwise, a motion vector and residue is interpolated from the base layer and applied to an decoded enhancement frame. More elaborate concealment, e.g., those that employ decoder search algorithms to obtain better motion correspondence, is also possible [3].

The concealment schemes above are typically applied to the location of loss, without detection and correction of subsequent drift. It is worth mentioning that the new *SP* and *SI* slice types of H.264 represent possible resynchronization points that can be reconstructed *perfectly* from one of multiple reference frames. They are complementary to our approach, and do not provide a measure of drift or allow correction of drifts in non *SP*/*SI* slices. The distributed source coding approaches such as [4] can sometimes provide eventual drift-free reconstruction in face of loss, but requires new coding tools and structures.

In this paper, we propose the additional use of the lower layer image as bounds for what a drift-afflicted higher layer frame could be, and perform *continuous* detecting and correcting for larger drift errors for improved video quality. The rest of this paper is organized as follows. An overview of our scheme is provided in Section 2.Evaluation results are given in Section 3 follow by a conclusion.

2. DRIFT DETECTION AND CORRECTION FOR SCALABLE VIDEO

For the sake of brevity and clarity, we assume a two-layer video with spatial scalability only. Extensions to more layers, and forms of scalability follows a similar argument. Let $\mathbf{f}$ and $\mathbf{F}$ be the decoded base and enhancement layer frames, respectively. Assuming no loss in base layer, we have,

$$\mathbf{f} = \mathbf{f}_e^c + \mathbf{f}_o$$

where $\mathbf{f}_o$ is the base layer original, and $\mathbf{f}_e^c$ is the loss-free compression noise. Similarly,

$$\mathbf{F} = \mathbf{F}_e^c + \mathbf{F}_e^d + \mathbf{F}_o$$

where $\mathbf{F}_e^d$ summarizes error from losses and drift. Only the quantities $\mathbf{f}$ and $\mathbf{F}$ are available at the decoder, and our goal is to construct from them a better approximation to $\mathbf{F}_o$ than $\mathbf{F}$ alone provides.

The basic strategy to decode only loss-free layers calls for simple upscaling:

$$\hat{\mathbf{F}}_o^{(1)} = up(\mathbf{f})$$

which, as we will show later in Section 3, may not be preferable to $\mathbf{F}$. A better approximation can be obtained as follows. Since the original image $\mathbf{f}_o$ is derived from $\mathbf{F}_o$ by image resizing, a loss-pass filter LP can be constructed such that

$$LP(\mathbf{F}_o) = up(\mathbf{f}_o)$$

Assume for the moment that distortions from compression are negligible, i.e., $\mathbf{F}_e^c = 0$ and $\mathbf{f}_e^c = 0$. Then,

$$\begin{aligned} LP(\mathbf{F}_e^d) &= LP(\mathbf{F}) - LP(\mathbf{F}_o) \\ &= LP(\mathbf{F}) - up(\mathbf{f}) \end{aligned}$$

which relates the observables $\mathbf{f}$ and $\mathbf{F}$ to the drift error $\mathbf{F}_e^d$ we wish to determine. Whether a pixel (i, j) in $\mathbf{F}$ is afflicted by drift can be determined by hypothesis testing. Specifically, we form the *consistency map* $\mathbf{C}$ by checking for each pixel (i, j) whether $|LP(\mathbf{F}_e^d)|$ is smaller than some threshold δ:

$$\mathbf{C}(i,j) = \begin{cases} 1 & \text{if } |LP(\mathbf{F})(i,j) - up(\mathbf{f})(i,j)| < \delta \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

A better approximate to the loss-free enhancement frame can be obtained by copying the parts of $\mathbf{F}$ that are consistent:

$$\hat{\mathbf{F}}_o^*(i,j) = \mathbf{C}(i,j)\mathbf{F}(i,j) + [1 - \mathbf{C}(i,j)]up(\mathbf{f})(i,j). \quad (2)$$

The condition in (1) explicitly poses a bound on which higher layer pixels can deviate from a base layer pixel, and is used for drift detection. Drift correction is performed in (2) by copying from $up(\mathbf{f})$. This frame $\hat{\mathbf{F}}_o^*$ is then taken as the final decoded frame for display and future reference.

Notice that an error signal $\mathbf{F}_e^d$ that resides entirely in the null space of the filter LP cannot be detected. In other words, the scheme makes an inherent assumption that drift errors have some low frequency components. Generally though, as errors propagate over time, it is unlikely that they always remain in the same frequency bands.

More generally when the compression noises are not negligible, it is easy to verify that:

$$LP(\mathbf{F}) - up(\mathbf{f}) = LP(\mathbf{F}_e^d) + LP(\mathbf{F}_e^c) + up(\mathbf{f}_e^c) \quad (3)$$

and the map $\mathbf{C}$ given by (1) would be inaccurate due to errors from compression noises. Nevertheless, the characteristics of compression noises are such that their energy distribution tend to sparse, e.g., concentrated near edges rather than uniform. As a result, one effective way to compensate for the error in $\mathbf{C}$ caused by compression noises is to remove small regions of inconsistency (smaller than 8 by 8 pixels) and only keep the larger regions. The final *despeckled* map

$$\mathbf{C}' = despeckle(\mathbf{C}) \quad (4)$$

is employed in (2) in place of $\mathbf{C}$. Since there is no way to determine if a small inconsistent region is caused by drift error, compression error, or both, the despeckling process can inadvertently cause small regions of significant drift error to be ignored. Nevertheless, the small spatial support of the speckle means the visual impact of such errors is small. Furthermore, if error propagation cause the error to grow in spatial support over time, it can be captured by the same drift detection and correction methods in (1) and (2).

Since compression errors are dependent on bit-rate or QP employed, it follows from (3) that the optimal δ for (1) should also be chosen according to QP, which is available to a decoder. Specifically, δ should be allow to change spatially according to changing QP, even though the simpler approach of using a constant $\delta = 9$ is taken in this paper.

In general, computational requirements for our proposed drift detection and correction methods are low since only simple filtering operations are required. Nevertheless, unlike typical error concealment method that are invoked only at the instance of loss, any drift detector needs to be in operation continously. Our techniques outlined in (1) to (4) requires only the decoded pixels (texture) in $\mathbf{f}$ and $\mathbf{F}$, and is generally applicable to all types of scalable compression methods. It should be noted however, that H.264 SVC are designed with a "single-loop decoding" feature [1, 2], that does not require generation of base-layer pixels when decoding at enhancement layer quality. As a result, extra decoding efforts are necessary to fully decode the base layer. Such overhead is not expected to be significant, especially when the base layer is of lower spatial resolution.

3. RESULTS

In this Section, we first illustrate the effectiveness of our drift detection and correction method by showing an example in which drift is allowed to build up, and then our technique is applied to one single frame only. We then show PSNR traces when continous drift detection and correction are in effect. Two sequences *Mobile Calendar* and *Jeff* of enhancement and base resolutions of 1280×704 and 320×176, respectively, are used, and are compressed using JSVM 9.9 software using a constant QP of 28. In all cases, only an intial I-frame is employed, and all subsequent frames are P-frames. The *Mobile Calendar* sequence is a challenging sequence with fine texture and complex motion, whereas *Jeff* is a typical video conference sequence with a large stationary background.

In all cases, the base layer is assumed to have been received losslessly, and losses in enhancement layers are concealed using *BLSkip* [2], which is found to be far superior to the other concealment methods available in JSVM.

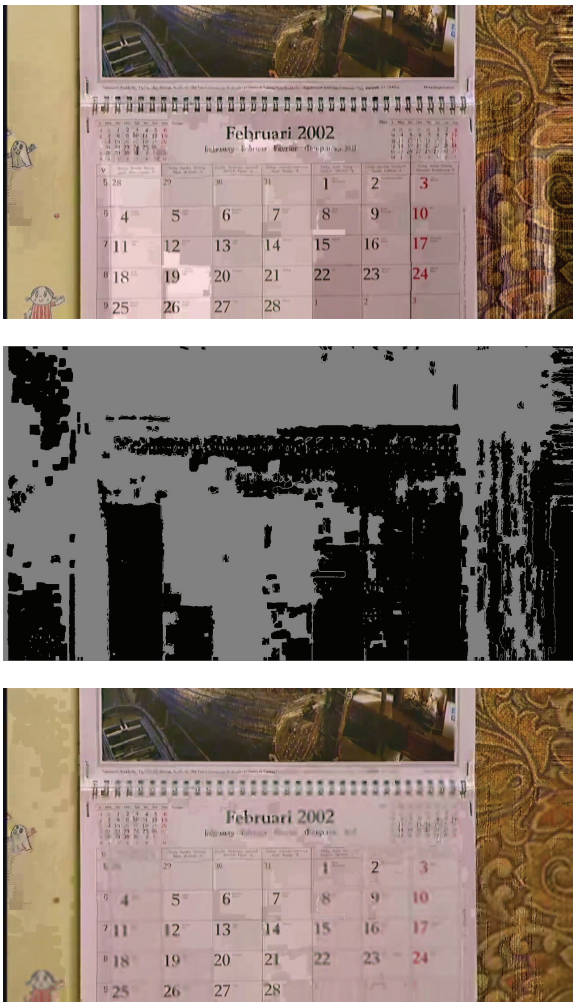

Fig. 1. Serious error develops from multiple earlier losses even after effective error concealment method *BLSkip* is applied for every loss (top). The corrupt frame is compared with base-layer reconstruction to obtain the despeckled consistency map (middle), where black indicates inconsistency. Finally, the corrected picture (bottom) is computed by selectively merging the consistent parts of the top frame. PSNR improves from 20.42 dB to 24.04 dB.

3.1. Single Frame Correction Example

In this example, the enhancement layer of *Mobile Calendar* sequence is subjected to 5% random frame loss, and our drift correction technique is applied to frame 160 only. The results are shown in Fig. 1.

First, we see that the bottom corrected picture is clearly much preferable to the drift-afflicted picture at the top. This is remarkable given that available "side information" or base-layer is only 1/16 the size of the full frame. We see that the significant color errors in both the wallpaper and calendar are corrected. Those errors originate from small imperfections in

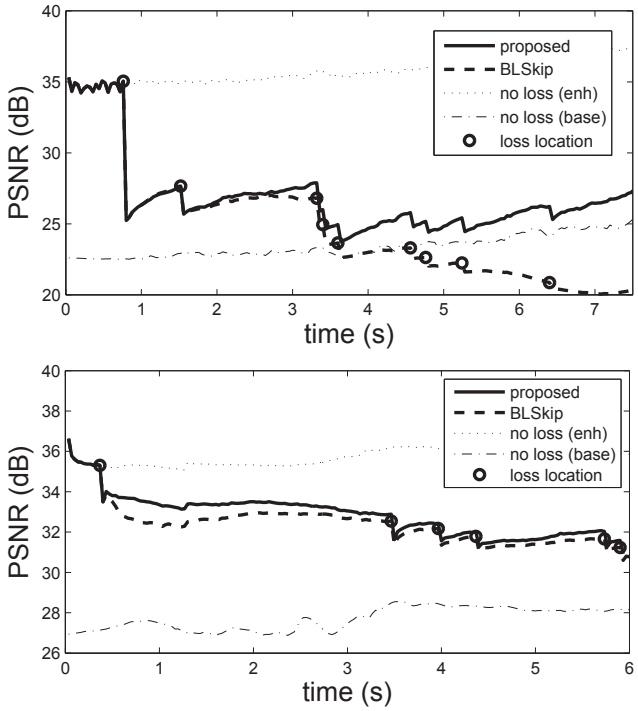

Fig. 2. PSNR traces for *Mobile Calendar* (top) and *Jeff* (bottom) when subjected to 5% enhancement frame loss.

error concealment that are continously copied as the calendar moves. These types of errors with large magnitude and area are readily corrected by our methods.

Next, we see that the corrected picture still contains non-uniform color in the calendar background, and some contours are visible. This is a direct consequence of the use of thresholding in (1). This may be corrected by changing (1) to use consistency probability, and is a subject of future research. We also see uncorrected scratches on the right part of the frame with wallpaper. Those are artifacts of the despeckling process, which cannot detect drift errors with small areas.

3.2. Continuous Drift Correction

We next apply continous drift correction to control drift for every frame so that unlike the exaple above, drift error never gets large before they are corrected. The comparison with and without continous drift correction for 5% enhancement frame loss rate is shown in Fig. 2 for both sequences *Mobile Calendar* and *Jeff*.

First, we see that the use of continuous drift correction never reduce quality. This is a consequence of our design to apply changes only when certain. Second, we see that the PSNR gain for continous drift correction can be large, exceeding 5 dB at places for *Mobile Calendar*. The PSNR gain for *Jeff* is more subdue because it contains a large stationary background, that any obtained gain are averaged over. Figs. 3

Fig. 3. Frame 160 of *Mobile Calendar* sequences without (top) and with (bottom) continuous drift correction. PSNR improves from 20.42 dB to 25.30 dB.

Fig. 4. Frame 40 of *Jeff* sequences without (top) and with (bottom) continuous drift correction. PSNR improves from 32.55 dB to 33.37 dB.

and 4 shows selected frames from the sequences. In particular, the corrected *Mobile Calendar* frame using continuous drift correction achieves 25.30 dB in PSNR, about 1.3 dB better than when the same technique is only applied once in Fig. 1. Generally, tradeoff between computation and quality can be achieved by not performing checking on every frame.

We notice that when continuous drift correction is not applied for *Mobile Calendar*, it is possible for the additional decoding of enhancement layer to result in a *lower* PSNR than discarding the lossy enhancement layer ("no loss (base)"). This suggest some quality indication, such as our drift detection, is necessary to guarantee positive quality return for decoding a lossy enhancement layer, even if such detection is only performed occasionally. In contrast, continous drift correction always perform better than dropping the enhancement layer.

Finally, we see that the PSNR obtained by "no loss (base)" is particularly low for *Jeff*, reaching a difference of 6 to 10 dB at places. This is again due to the large stationary background that is easily concealed, but whose fine textures are discarded when the enhancement layer is dropped. This argues against the dropping of enhancement layers even when they are lossy.

4. CONCLUSION

In this paper, we introduce a simple yet effective drift detection and correction method for scalable video, and show via experiments that PSNR gains of up to 5 dB can be obtained for H.264 SVC video. The approach does not require additional bit overhead or changes to video compression method.

There are a number of possible extensions, including more general detection than given by the binary decisions of (1), and better correction method than given by (2), e.g., by using decoder motion search as in [3], or by fusing different frequencies from $\mathbf{f}$ and $\mathbf{F}$ [5].

5. REFERENCES

[1] H. Schwarz, Detlev Marpe, and T. Wiegand, "Overview of the scalable video coding extension of the H.264/AVC standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 9, pp. 1103–1120, September 2007.

[2] Y. Guo, Y. Chen, Y. Wang, H. Li, M. Hannuksela, and M. Gabbouj, "Error resilient coding and error concealment in scalable video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 6, pp. 781–795, June 2009.

[3] C. Yeo, W. Tan, and D. Mukherjee, "Receiver error concealment using acknowledge preview (recap) - an approach to resilient video streaming," in *Proceedings ICASSP*. IEEE, April 2009, pp. 785–788.

[4] A. Sehgal, A. Jagmohan, and N. Ahuja, "Wyner-ziv coding of video: an error-resilient compression framework," *IEEE Trans. Multimedia*, vol. 6, no. 2, pp. 249–258, April 2004.

[5] F. Brandi, R. Queiroz, and D. Mukherjee, "Super-resolution of video using key frames and motion estimation," in *Proceedings ICIP*, October 2008.