Low Bitrate Watercolor Video

Keman Yu
Jiang Li
Jizheng Xu
Shipeng Li

August 2002

Technical Report
MSR-TR-2002-88

Microsoft Research
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052

Low Bitrate Watercolor Video

Keman Yu, Jiang Li, Jizheng Xu, and Shipeng Li
Microsoft Research Asia
3F Sigma Building, 49 Zhichun Road
Beijing 100080, China
+86 10 62617711-3115
{i-kmyu, jiangli, spli}@microsoft.com

ABSTRACT

We developed a low bitrate watercolor-like video form, which is constructed with a bi-level Y luminance component and a number of UV chrominance combinations. The bi-level Y-component outlines the features of an image and the UV-combinations describe the basic color information of the image. The UV combinations are dynamically chosen during encoding. The resultant video looks better than bi-level video and its bitrate is still lower than that of full-color video. Watercolor video can be used in video broadcast and communication in low bandwidth networks.

1. INTRODUCTION

Network services are being developed towards allowing users to access information anywhere, anytime on any device. Video as an important part of information or media is required to meet this demand. While it is relatively easy to transmit popular MPEG [1] or H.26x [2] video through broadband, it is not so easy to transmit such kinds of videos through wireless networks with limited bandwidths. The bottleneck of transmitting videos through any network is usually in the low end.

As a solution to transmitting video under low bandwidth network conditions, we previously proposed a video form, called bi-level video [3]. Bi-level video is generated by converting a gray-scale image to a bi-level image and then compressing a bi-level image sequence into a bi-level video. Special considerations are taken in the threshold and compression processes so that the resultant videos possess low bitrate, clear shape, smooth motion, and low latency properties.

Although bi-level video achieves very low bitrate, its appearance is only black and white and therefore is still not satisfactory. As we know, in MPEG video, the UV components actually only occupy about 5-10% of the total bitrate cost. Can we just add a limited number of UV components to bi-level video so that the whole visual quality is improved but the bitrate can still be low?

Yes. The types of colors in a low resolution CIF or QCIF-size image are actually limited. By converting a gray-scale image to a bi-level image, we reduced the complexity of the Y component but preserved the most important outline features if the threshold is selected suitably. The addition of UV components will assign more color features to white pixels in the previous bi-level images. As shown in Fig.1, the appearance of such kinds of videos is similar to a watercolor -- a paint composed of a water-soluble pigment, which displays basic colors of a picture (please refer to the electronic file for a better view).

(a) (b) (c)

Figure 1: (a) Full-color image, (b) watercolor image and (c) bi-level image.

In section 2, we will describe the compression algorithm of watercolor video. The algorithm is assumed to be first applied to the I-frames of a video and then we will describe how to deal with the P-frames. Experiments on test video clips are shown in section 3. We will conclude the work and point out future directions in section 4.

2. ALGORITHM

The key technologies in watercolor video coding are how to describe the chrominance information of an image and how to select representative chrominance information and compress it.

2.1. Which Kinds of UV Components Are Needed?

Our design of watercolor video is based on our valuable experience in both bi-level video coding and MPEG video coding. Basically, in MPEG video coding, each picture is divided into some blocks. Each block is usually 16x16 in Y and 8x8 in U and V respectively. The resolution of the U and V components is usually half of that of Y. These Y, U, and V values in each block are transformed from the spatial domain into a set of DCT coefficients of the frequency domain. Each of these coefficients is a weight associated with its corresponding DCT basis waveform.

These coefficients are then quantized, and nonzero quantized values are compressed using an entropy coder. As we only use a bi-level Y component, we also need not use very accurate UV distributions. Experimental results shown in the latter part of this paper will justify this consideration. Obviously, the most basic part of the DCT coefficients of each U and V block is just the first coefficient, i.e. the DC component. The physical meaning of the DC component is the average chrominance of the block.

2.2. How Many Typical UV Combinations Are Needed?

As watercolor video is specially developed for low bitrate video communication, we can only consider small resolution images. Take a QCIF-sized image for example. If we divide a QCIF-sized (176×144) image into 16x16 blocks, the total number of such blocks is 99. So there are totally 99 UV combinations after the above averaging process. If we just use these 99 UV combinations, the resultant data will be very diverse. In order to group these UV combinations and obtain a limited number of representative UV combinations, we reassign the values of U and V to the nearest multiple of 4. After that, the total number of UV combinations is reduced to about 20-30 according to experiments that we have done on many video sequences. These UV combinations can be represented by indexes. However, it is not necessary to set so many indexes. As 4 bits can be used to represent 16 indexes, we setup totally 16 indexes to represent the most frequent 16 UV combinations. For the remaining UV combinations, their 2-dimensional Euclidean distances from each of the 16 typical UV combinations are calculated, and each of the remaining values is merged to the nearest typical UV combination.

2.3. Compressing the UV Components

Now, we have a UV combination lookup table which stores the U and V values of 16 typical UV combinations of an image. Each 16x16 block of an image is assigned with an index of the table. Huffman coding is employed to encode these indexes. We calculate the probability for each entry of the lookup table in terms of its frequency and make binary codes from probabilities using Huffman coding. For example, when we compress the indexes of blocks in the QCIF Akiyo video sequence, we get an average code length of 3.86 bits, therefore the UV components of each frame occupy only 99×3.86 = 382 bits.

2.4. P-frame consideration

The above procedure is assumed to be first applied to the I-frames of a video sequence. For P-frames, the averaged UV components of each block are classified to the typical UV combinations of I-frames. This treatment will not cause obvious errors because in a low bitrate video communication scenario, people are not very sensitive to details of a small scene, and the background is usually not changed. The central part of the scene is usually the user's face which possesses only limited color types. Since we can use the same lookup table of an I-frame for its subsequent P-frames, we need only transmit the lookup table which stores the U and V values of the typical UV combinations of each I-frame.

2.5. Decoding and Displaying

The bit stream of each frame of a video is composed of a bit stream of a bi-level Y component and a bit stream of UV components. In the decoder, the bit stream of the bi-level Y component and the bit stream of the UV components are decompressed and output to display. As we have mentioned above, block artifacts usually occur since we only use the average UV values of a 16x16 block. On the display side, we use bilinear interpolation to eliminate them. For pixels that are located along the edge of the image, their UV values are just assigned with the UV values of the block.

2.6. Optimizing the Selection of Typical UV Combinations

As we have described above, the lookup table is composed of 16 combinations with the highest statistical probabilities. If the chrominance of an image is distributed relatively uniformly, i.e. the differences of the probabilities of these UV combinations are relatively small, this method will lead to evident errors. In order to overcome this drawback, we adopt Lloyd-Max Quantization theory [4] to obtain the optimal UV combinations through iterations. The procedure is as follows:

(1) Reassign the U and V values of each UV combination to the nearest multiple of 4.
(2) Select the 16 most frequent UV combinations.
(3) All the other UV combinations are merged to the closest entry according to their 2-dimensional Euclidean distances.
(4) Calculate the weighted average of the U and V values of each entry.
(5) If the U and V values of the 16 UV combinations no longer change, or the iteration reaches a maximal number, the iterations stop; otherwise go to step (3).

2.7. Further Optimizing the UV Combinations

In the display side, the bi-level Y component and UV components are combined to produce the final color. For those UV combinations that are generated from blocks in

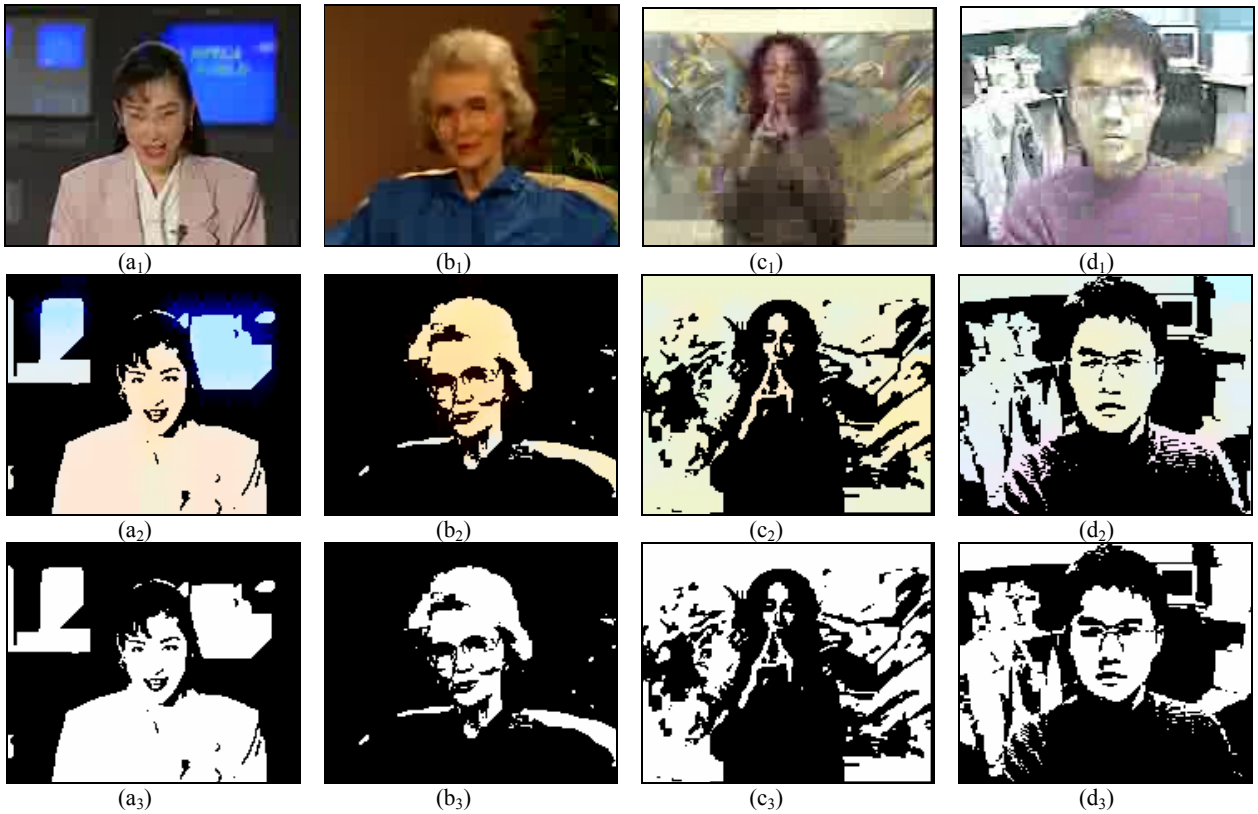


Figure 2: One typical frame of each clip coded at 9.6 Kbps.

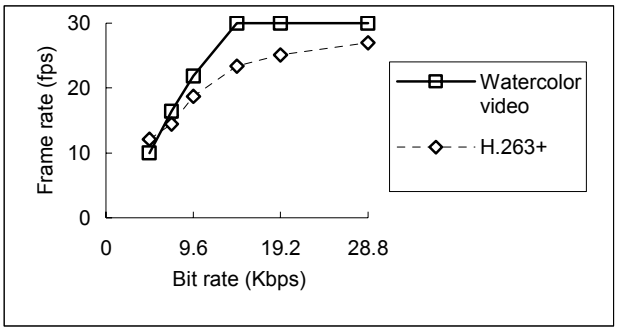
which the Y component value is 0, the final color in the display is also black. In other words, they do not affect the appearance of the watercolor image. So UV combinations that are only generated from Y=0 blocks should be ignored in the statistical process. This results in more indexes being assigned to UV combinations that visually affect the watercolor image. In practice, it is hard to find a block in an image where all the Y component values are 0, so we disregard the UV combination of a block if over 90% of the Y component values are 0. Since as described in subsection 2.4, the scene in P-frames is very similar to that of the I-frame, this method will not obviously affect the quality of P-frames. Experiments on the QCIF Akiyo video clip show that the final average code length of each block is reduced to 2.28 bits with this method.

3. EXPERIMENTAL RESULTS

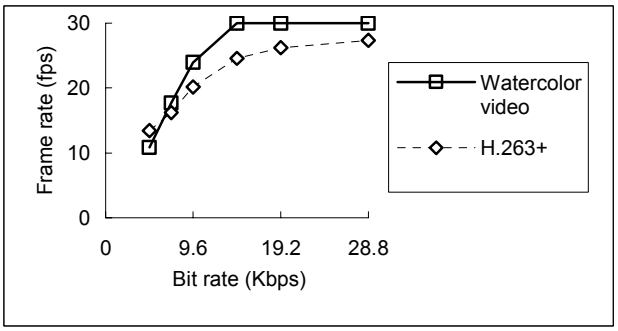
We tested our approach on both standard MPEG test video clips (Fig.2 (a), (b), (c)) and ordinary clips captured from real scenes using PC digital cameras (Fig.2 (d)). Video clip (a) Akiyo is representative of scenes with little head motion, clip (b) Grandma typifies scenes with relatively large head motion and clip (c) Silent possesses large motion of arms and hands. Video clip (d) Yu represents scenes with very large motion of the whole body. All the video clips are in QCIF format at a frame rate of 30 fps. We compared the visual effects and frame rates obtained by our method and one of the H.263+ implementations [5].

Fig.2 (a) through (d) show one typical frame of each clip coded with the H.263+ encoder, the watercolor video encoder and the bi-level video encoder respectively. It is shown that the images (with subscript 1) coded with the H.263+ encoder are filled with some blocks which result in the artifacts on Akiyo, Grandma and Yu's faces and Silent's whole body, while the images (with subscript 2) coded by the watercolor video encoder are as clear as the images (with subscript 3) coded by the bi-level video encoder. Furthermore, the basic colors of the original clips are kept and the whole visual effect is obviously better than that of the bi-level images.

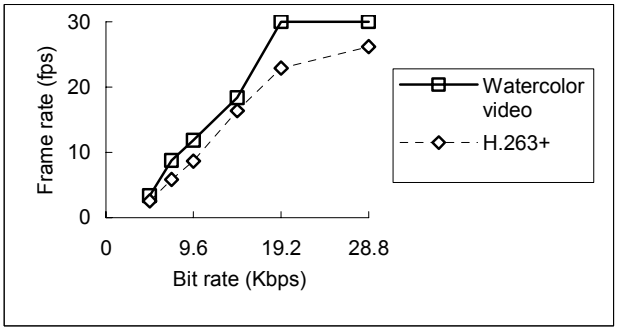
The smoothness of motion is a feature of watercolor video. Fig.3 (a) through (d) show the frame rates of each clip coded by the watercolor video encoder and the H.263+ encoder. The frame rates of coded video clips are counted when bandwidths are set to 4.8, 7.2, 9.6, 14.4, 19.2 and 28.8 Kbps, and the I-frame interval is set to 5 seconds. Experiments show that in most cases the watercolor video encoder generates higher frame rates than the H.263+ encoder. Therefore, watercolor video provides smoother motions of scenes. The exceptions are that in clip (a) Akiyo and (b) Grandma, the frame rates generated by H.263+ encoder are a little higher than those produced by the watercolor video encoder when the bandwidths are



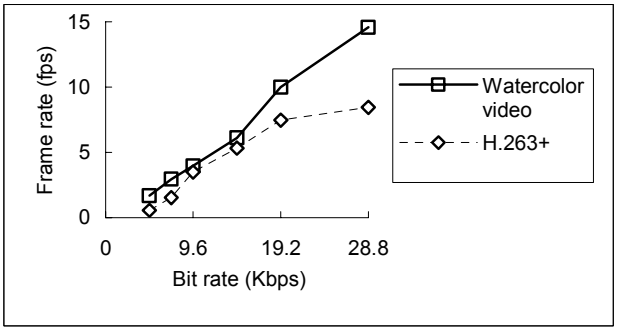
(a) Akiyo



(b) Grandma



(c) Silent



(d) Yu

Figure 3: Frame rates of each clip code by the watercolor video encoder and the H.263+ encoder.

about 4.8 Kbps. This is because with such bandwidths, even if we removed the chrominance information from the watercolor video, the frame rates generated by the two encoders are very close. However, given the same conditions in clip (c) Silent and (d) Yu, the large motion made the frame rates produced by H.263+ encoder lower than those obtained by the watercolor video encoder.

4. CONCLUSIOINS

We have developed a low bitrate watercolor video form. For an input video, its Y component is compressed using bi-level video coding. The UV components are first averaged in each block, and then typical UV combinations are selected according to iterative grouping results of these combinations. After that, the chrominance information of each block is represented by an index to a lookup table of typical UV combinations. These indexes are further encoded using Huffman coding. The lookup table is transmitted with each I-frame. For P-frames, no selection process is needed, and the chrominance information of each block is just classified to the typical UV combinations of its corresponding I-frame.

Experiments on standard MPEG test video clips and captured clips of ordinary scenes show that watercolor video looks better than bi-level video, and its frame rate is still higher than that of full-color video under the same bandwidth conditions.

Future research directions could include how to establish a scalable coding method of chrominance and how to balance the bitrate cost in the Y component and the UV component to therefore reach the best quality for any given bandwidth.

5. REFERENCES

[1] ISO/IEC JTC1/SC29/WG11 N3312 Coding of moving pictures and audio March 2000/Noordwijkerhout.

[2] ITU-T Recommendation H.263 Video coding for low bit rate communication, 02/98.

[3] Jiang Li, Gang Chen, Jizheng Xu, Yong Wang, Hanning Zhou, Keman Yu, King To Ng and Heung-Yeung Shum, "Bi-level Video: Video Communication at Very Low Bit Rates," ACM Multimedia Conference 2001, September 30 – October 5, Ottawa, Ontario, Canada, pages 392-400.

[4] A. K. Jain, Fundamental of Digital Image Processing, Prentice Hall, 1989.

[5] H.263+ public domain code, TMN Version 3.0, Dept of Electrical Engineering, University of British Columbia, Canada, http://spmg.ece.ubc.ca/h263plus/h263plus.htm