Scalable Portrait Video for Mobile Video Conferencing

Jiang Li
Keman Yu
Tielin He
Yunfeng Lin
Shipeng Li

July 2002

Technical Report
MSR-TR-2002-79

Microsoft Research
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052

Scalable Portrait Video for Mobile Video Conferencing

Jiang Li, Keman Yu, Tielin He, Yunfeng Lin, and Shipeng Li

Microsoft Research Asia
3F Sigma Building, 49 Zhichun Road
Beijing 100080, China
+86 10 62617711-5678

{jiangli, i-kmyu, spli}@microsoft.com

ABSTRACT

Wireless networks have been rapidly developing in recent years. GPRS and CDMA 1X for wide areas, and 802.11 and Bluetooth for local areas have already emerged. Broadband wireless networks urgently call for rich contents for consumers. Among various possible applications, video conferencing is one of the most promising for mobile devices on wireless networks. This paper describes the generation, coding and transmission of an effective video form, scalable portrait video for mobile video conferencing. As an expansion to bi-level video, portrait video is composed of more gray levels, and therefore possesses higher visual quality while it maintains a low bit rate and low computational costs. Portrait video is a scalable video in that each video with a higher level always contains all the information of the video with a lower level. The bandwidths of 2-4 level portrait videos fit into the bandwidth range of 20-40 Kbps that GPRS and CDMA 1X can stably provide; therefore, portrait video is very promising for video broadcast and communication on 2.5G wireless networks. With portrait video technology, we are the first to enable two-way video conferencing on Pocket PCs and Handheld PCs.

Categories and Subject Descriptors

1[**Compression**]

General Terms

Algorithms, Performance, Design, Experimentation, Theory

Keywords

Portrait video, bi-level video, video coding, mobile video conferencing, video broadcast and communication

1. INTRODUCTION

Wireless networks have been deployed rapidly in recent years. GPRS (General Packet Radio Service) and CDMA 1X (Code Division Multiple Access) as 2.5G solutions to wide area wireless networks are available in increasingly more regions in Europe, North America and Southeast Asia. Wireless LAN 802.11 and Bluetooth also compete strongly for local area wireless networks. The fast expansion of wireless networks calls for rich contents and services for consumers. However, due to limited channel bandwidths in these wireless networks and weak processing power in mobile devices, conventional media contents have difficulties in distribution.

Bi-level video [1] is an effective solution for low bandwidth mobile video conferencing, where previously there did not exist suitable video coding technology for current wireless network and mobile device conditions. We observed that although conventional video processing and coding technologies such as MPEG1/2/4 [2] and H.261/263 [3, 4] could also code video for low bit rates, the resultant images usually looked like a collection of color blocks and the motion in the scene became discontinuous. The block artifacts of these methods originate from the common architecture of MPEG1/2/4 and H.261/263, i.e. discrete cosine transform (DCT) based coding. In DCT based coding, low spatial frequency values that represent the "basic colors" of the blocks possess high priority. However, in video communications, facial expressions that are represented by the motions of the outlines of the face, eyes, eyebrows and mouth deliver more information than the basic colors of the face. Bi-level video uses bi-level images to represent these facial expressions, which results in very high compression ratios. Experiments show that in low bandwidths, bi-level video provides clearer shape, smoother motion, shorter initial latency and much cheaper computational cost than do DCT-based technologies. Bi-level video is especially suitable for small mobile devices such as handheld PCs, palm-size PCs and mobile phones that possess small display screens and light computational power, and that work in wireless networks with limited bandwidths.

In bi-level video, scenes are always represented by two colors, usually black and white. Although black and white colors are sufficient to describe the outlines of a scene, their visual quality is not very satisfactory. Given that many mobile devices are now able to display at least four levels of grayscale, users of our earlier versions of Microsoft Portrait, a research prototype for mobile video conferencing, have expressed a desire for improved video that contains more gray values and has better visual quality. With this improved video, bit rates must also be kept low. This is exactly the problem we aim to solve in this paper.

After reviewing existing video technologies that cover different bandwidth ranges, we find that MPEG/H.26x performs well in the

bandwidth range greater than about 40 Kbps and bi-level video works well in the range of 10-20 Kbps for QCIF size. However, the visual quality of bi-level video could no longer be improved even if greater bandwidth is assigned to it. Now the task is how to improve the visual quality of bi-level video in the bandwidth range of 20-40 Kbps or how to design a new video form that can work well in this range. It is very important to develop a video form to fit into the 20-40 Kbps bandwidth range because this is the range that 2.5G wireless networks such as GPRS and CDMA 1X can stably provide, although the theoretical bandwidths of GPRS and CDMA 1X are 115 Kbps and 153.6 Kbps, respectively.

We will present the general architecture of the generation, coding and transmission of scalable portrait video in Section 2. The thresholding algorithms that are used to generate multi-level image sequences are described in Section 3. Section 4 and Section 5 depicts the coding of four-level video and multi-level video, respectively, and the transmission of multi-level video is detailed in Section 6. Experimental results are displayed in each of these sections to accompany the description of the algorithms. We will summarize the features of scalable portrait video and discuss future directions in Section 7.

2. ARCHITECTURE

A better approach to providing a video technology that works well in both the 10-20 Kbps and 20-40 Kbps bandwidth ranges is to propose a scalable video technology. Our proposal is that if the given bandwidth is about 10-20 Kbps, then the previous bi-level video is employed; if the available bandwidth increases to about 20-40 Kbps, then more grayscale levels are added to the previous bi-level video and the transition process will be smooth. The final result is that greater bandwidths are utilized with more grayscale levels displayed.

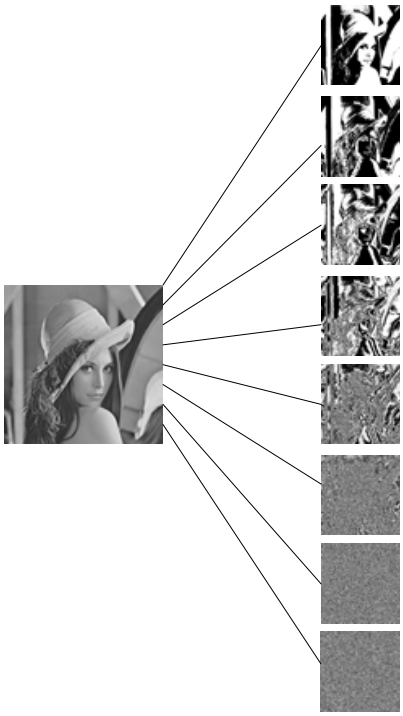

Figure 1: A grayscale image (Lena) and each of its bit planes.

An intuitive solution to getting more grayscale levels into video is to code each bit plane of a grayscale video in turn. Fig. 1 shows a grayscale image and its bit planes from the highest bit plane at the top to the lowest bit plane at the bottom. As we can see, in those lower bit planes, the images appear like noise. It is very difficult to compress such noisy images. Of course, the efficiency of the approach in which we code all the bit planes separately for a full grayscale image sequence will be much lower than that of a DCT based coder, in which only residuals of an image sequence are coded. We will compare multi-level video coding and DCT based coding in Section 5. The bit-plane approach is applicable only if we apply it to limited numbers of bit planes.

Another solution is based on multiple divisions. If we use a threshold to convert a grayscale image (Fig.2 (a)) into a bi-level image (Fig.2 (d)), the threshold actually divides the original grayscale image into two parts: one part consisting of all the pixels with grayscale values greater than or equal to the threshold (Fig.2 (b)), and the other part consisting of all the pixels with grayscale values less than the threshold (Fig.2 (c)). Their histograms are shown at the left. The remaining areas in Fig.2 (b) and (c) are marked by red (please refer to the pdf file for a better view). If 2 bits are available for each pixel of the final image, we could further detail both the brighter and darker areas of the bi-level image (Fig.2 (d)) with two gray levels respectively. This can be realized by using a higher threshold to divide Fig.2 (b) into two parts: Fig.2 (e) and Fig.2 (f), and using a lower threshold to divide Fig.2 (c) into two parts: Fig.2 (g) and Fig.2 (h). The remaining areas in Fig.2 (e), (f), (g) and (h) are also filled by red. The resultant bi-level images are Fig.2 (i) and Fig.2 (j), respectively. The first bit plane of the four-level image (Fig. 2(l)) is exactly the bi-level image Fig. 2(d). In the second bit plane (Fig. 2(k)) of the four-level image, for each pixel that is located in the brighter area of Fig. 2(d), its second bit value is equal to the binary value of the pixel in Fig. 2(i), and for each pixel that is located in the darker area of Fig. 2(d), its second bit value equals the binary value of the pixel in Fig. 2(j).

After reviewing the traditional 256-level grayscale representation of a grayscale image, we find that it is actually the result of a procedure that divides the image with a threshold of 128, and then further divides the divided parts of the image with thresholds of 192 and 64 respectively, and so on. The difference between the traditional 256-level grayscale representation of an image and our bi-level and four-level representations is that the traditional representation always uses 128 as the threshold of the first division, 192 and 64 as the thresholds of the second division, and so on, but our methods may not. It is obvious that these threshold values may not be optimal. The first problem we need to solve in this paper is that if we are allowed to represent an image with 1 bit plane, how we can choose the most suitable threshold to divide the original image into two parts that result in a good bi-level image; and if we are allowed to represent an image with 2 bit planes, how you can choose three suitable thresholds to divide the image into four parts that result in a good four-level image, and so on.

Back to the compression of multiple bit planes of an image, the noise in the lower bit planes is the major obstacle in compression. An image can be well compressed if it is regular. In arithmetic coding, if the value of a pixel can be predicted well, it can be coded well. The prediction quality of a pixel value depends on the regular distribution of its context, i.e. its neighbor pixels.

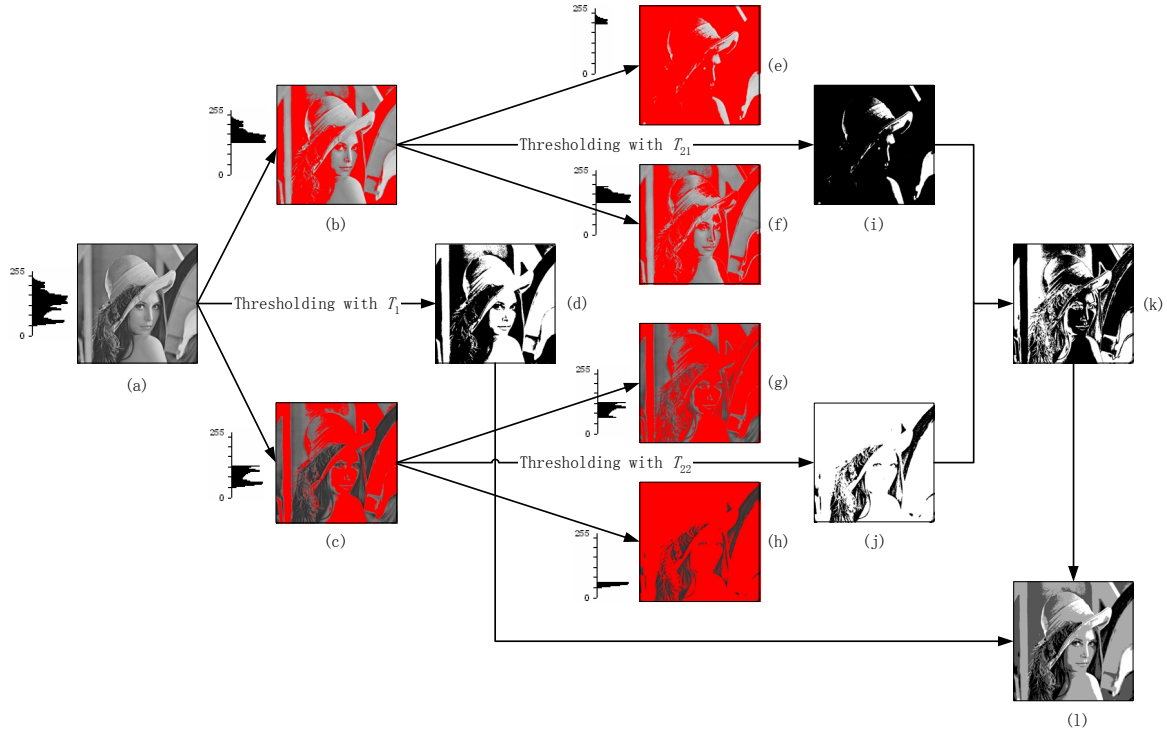

Figure 2: The generation of a four-level image.

As described previously, the second bit plane (e.g. Fig. 2(k)) of a four-level video consists of two parts: one part consisting of pixels of the bi-level image (Fig. 2(i)) generated using the higher threshold; the other part consisting of pixels of the bi-level image (Fig. 2(j)) generated using the lower threshold. Which pixel of the two bi-level images contributes to the second bit plane is determined by the binary value of the pixel in the first bit plane (Fig. 2(d)). Although the second bit plane (Fig. 2(k)) appears complex, its components, Fig. 2(i) and Fig. 2(j) are relatively simple. They are actually bi-level images that are converted from the original image with a higher threshold and a lower threshold respectively. So we can see that the coding of the low bit plane of a four-level video is just the combination of the coding of two mutually exclusive bi-level video parts with their thresholds to be chosen at a higher and a lower values respectively. In addition, we do not need to spend extra bits to describe the regions of the video parts since they are defined by the first bit plane of the four-level video. This means that the coding of the low bit plane of a four-level video is as easy as the coding of the high bit plane. The same principles can be extended to more bit planes of a multi-level video.

The advantage of the above multi-level video coding is that the transmission of more bit planes always results in the enhancement of the visual quality of the lower level video. For example, if we use a bi-level video to start, then we want to switch to four-level video as more bandwidth is available, we need only to keep the bi-level video coding for the first bit plane and start coding the second bit plane for the four-level video. If the client receives the bit stream of the second bit plane of the four-level video, then users can see the brighter part of the original bi-level video as having two grayscale levels, as does the darker part. If the bit stream of the second bit plane is lost during transmission, the client can still display a bi-level video using the bit stream of the first bit plane of the four-level video. Moreover, since the first bit plane of a four-level video is just the original bi-level video, we can fade in the second bit plane so that a smooth transition can be reached. The same fading method can be extended to switching between arbitrary levels of video. The property that a video with a higher level always contains all the information of a video with a lower level exactly indicates scalability of the video.

3. How to Express an Image into Given Gray Levels

One of the core problems in multi-level video generation is how to express an image using given gray levels. The problem is meaningless if the target number of gray levels approaches 256, i.e. full grayscale levels. However, if the number of gray levels is about 2, or 4, the effects are very obvious as shown in Fig 3. Besides bandwidth considerations, this issue also has practical impact in that many mobile phone screens are in 4-level grayscales.

(a) (b)

Figure 3: 4-level grayscale images (Carphone) without (a) and with (b) optimal thresholds

The basic approach to expressing an image using given grayscale levels is thresholding. If we want to convert a grayscale image into a bi-level image, we need only one threshold. If we want to express an image into multiple levels, we need multiple thresholds.

Image thresholding methods are popular techniques in image segmentation. They can be classified into nonparametric approaches (usually solving an optimal problem on some criteria) and parametric approaches, based on the mathematical methods they use. Among methods that use only the gray-level distribution of an image, Ostu's Between-class variance method [5], Entropy methods [6, 7], and Minimum error method [8] are typical nonparametric methods, and Moment-preserving method [9] is typical parametric method. The most famous methods that use local spatial features are two-dimensional thresholding methods [10, 11] and multi-dimension method [12]. The thresholding methods utilizing local spatial features are too time-consuming to be used in real time applications. We adopt Ostu's Between-class variance method [5] because it is superior in its uniformity measure and shape measure for most types of images [13]. Ostu's method chooses the optimal thresholding values to maximize variances between gray level classes.

Suppose we want to convert an image with L (usually 256) gray levels into M (< L) gray-levels. Let b_1, b_2 … b_M be the principal gray-levels and let $f(x, y)$ denote the gray level of the pixel in position (x, y). If the multi-threshold value vector is $(T_1 \ldots T_{M-1})$, the multi-thresholding procedure is as follows:

$$f_T(x,y) = b_i, \quad \text{if} \quad f(x,y) \in C_i, \quad i = 1, 2, \ldots M. \tag{1}$$

where the M gray level ranges are:

$$C_1 = [1,...,T_1 - 1], \tag{2}$$

$$\ldots$$

$$C_i = [T_{i-1},..,T_i - 1],$$

$$\ldots$$

$$C_M = [T_{M-1},...,L]$$

The probability of the gray level i is:

$$p_i = \frac{f_i}{N}, \quad i = 1, \ldots, L \tag{3}$$

where f_i is the frequency that the gray level i occurs in the image, N is the total number of pixels in the image. So the probability distributions for M gray level classes are $\frac{p_j}{\omega(i)}$, where

$$\omega(i) = \sum p_j, \quad j \in C_i, \quad i = 1, \ldots M \tag{4}$$

The between-class variance of the M classes is defined using discriminant analysis:

$$\sigma_B^2 = \sum_{i=1}^{M} \omega(i)(\mu(i) - \mu_T)^2, \tag{5}$$

where the total gray level expectation

$$\mu_T = \sum_{i=1}^{L} i p_i, \tag{6}$$

and the ith gray level range expectation

$$\mu(i) = \sum j \frac{p_j}{\omega(j)}, \quad j \in C_i \tag{7}$$

The optimal thresholds vector is selected by maximizing σ_B^2

$$(T_1 \ldots T_{M-1}) = \text{Arg Max}\{\sigma_B^2\} \tag{8}$$

The time consumed for exhaustive search in the threshold vector space increases as $O(L^{M-1})$. So the multi-thresholding method is very time consuming if M is large. Liao et al. [14] utilized a recursive algorithm and a look up table to accelerate Ostu's method and has avoided a large amount of repeated calculations.

(a) (b) (c)

Figure 4: One frame of a full grayscale video and its bi-level and four-level images generated using Ostu's method.

Fig. 4 shows one frame of a full gray scale video (Salesman) and its bi-level and four-level images generated using Ostu's method. For the sake of scalability, in four-level video, instead of using Ostu's multiple thresholding method, we recursively use Ostu's single thresholding method on the image and its divided parts, i.e. we first use Ostu's single thresholding method on the whole image to get a threshold for the generation of the first bit plane, then apply Ostu's single thresholding method to two divided parts of the image respectively. For 2^i level video (where $i \geq 3$), thresholds are determined by equal subdivision beginning with the third bit plane. The above method as an automatic threshold generation method is useful for the conversion of large amounts of video clips into a given number of grayscale levels.

(a) (b)

Figure 5: Two four-level images generated using Ostu's method (a) and our empirical method (b).

In video communications, although the threshold calculated by an automatic thresholding method may be optimal to the whole scene, it may not be optimal to the face that we are paying most of our attention to. In these cases, we allow users to fine tune the threshold. While it is easy for a user to adjust one threshold in a bi-level video, we show that the user can also adjust multiple thresholds in a multiple-level video. For bi-level videos, we certainly let users adjust one threshold. For four-level videos, after a large amount of experiments on faces of different races and with different lighting conditions, we found an empirical method. The method is that we let users adjust the threshold of the first bit plane, called the principal threshold, of a four-level video, and then the higher threshold and the lower threshold for the second

bit plane are always set as the principal threshold plus and minus 16 respectively. As shown in Fig. 5, the visual quality of a four-level image that is generated using our empirical method (b) is better than that generated using Ostu's method (a).

4. Four-level Video Coding

Four-level video coding is actually a combination of bi-level video coding. Let us first briefly review the coding process of bi-level video (please refer to [1] for details). In the coding stage of a bi-level video, each binary pixel is coded in raster order. The process for encoding a given pixel is: (a) computing a context number; (b) indexing a probability table using the context number; and (c) using the indexed probability to drive an arithmetic coder. When the final pixel has been processed, the arithmetic code is terminated.

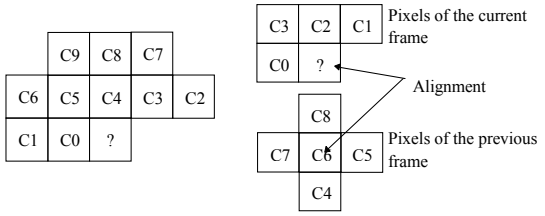

Figure 6: (a) The intra template and context construction. (b) The inter template and context construction. The pixel to be coded is marked with '?'.

Each frame can be compressed with a context-based arithmetic encoder (CAE) in intra or inter mode. Both modes result in the generation of a single binary arithmetic codeword. The various coding modes are characterized by their context computation and the probability table used.

In this module, the same template and context construction scheme as those in MPEG4 are applied. In detail, for intra coded frames, a 10-bit context $C = \sum_k c_k \cdot 2^k$ is built for each pixel as illustrated in Fig. 6(a). And for inter coded frames, temporal redundancy is exploited by using pixels from the previous frame to make up part of the context. Specifically, a 9-bit context $C = \sum_k c_k \cdot 2^k$ is built as illustrated in Fig. 6(b). When building contexts, any pixels to the left of or above the bounding box of the current frame are assumed to be zero.

In the coding of four-level video, the coding of the first bit plane (as in Fig. 2(d)) is just the same as the coding of a bi-level image sequence. The threshold is chosen using methods described in Section 3. As indicated previously, the coding of the second bit plane (as in Fig. 2(k)) is the combination of the coding of two bi-level image sequences (as in Fig. 2(i) and Fig. 2(j)) that are generated from the original image with a higher threshold and a lower threshold respectively. Suppose T_1 is the threshold for the first bit plane, i.e. the principal threshold. T_1 divides the grayscale image $f(x, y)$ into two parts: S_1 and S_2

$$S_1 = \{(x, y) \mid f(x, y) \geq T_1\} \quad (9)$$

and

$$S_2 = \{(x, y) \mid f(x, y) < T_1\} \quad (10)$$

Suppose T_{21} is the threshold applied to S_1 for the generation of part of the second bit plane of the four-level video, and T_{22} is the threshold applied to S_2 for the generation of the other part of the second bit plane. Assume that $b(x, y)_i$ is an array of two-bit elements that is used to record the resultant four-level image where $i = 1$ corresponds to the high bit and $i = 2$ corresponds to the low bit. The feature of the coding of the second bit plane $b(x, y)_2$ of a four-level video is that when a pixel (x, y) is coded, we need to first determine whether it belongs to S_1 or S_2. Its context for the intra or inter template should consist of pixels in the corresponding bi-level image (as in Fig. 2(i)) of S_1 or the corresponding bi-level image (as in Fig. 2(j)) of S_2. Only by this can we make the compression of the second bit plane of a four-level video as simple as that of the first bit plane. The algorithm for calculating the context of the intra or inter template of pixel (x, y) is as follows:

If $b(x, y)_1 = 1$, i.e. $f(x, y) \geq T_1$, i.e. $(x, y) \in S_1$
  For each pixel (u, v) in the intra or inter template of (x, y)
    If $b(u, v)_1 = 1$
      The binary value of (u, v) in the template $c(u, v) = b(u, v)_2$
    Else
      $c(u, v) = 0$
    End if
  End for
Else
  For each pixel (u, v) in the intra or inter template of (x, y)
    If $b(u, v)_1 = 0$
      $c(u, v) = b(u, v)_2$
    Else
      $c(u, v) = 1$
    End if
  End for
End if

As an effective bit rate reduction method in bi-level video coding, a threshold band is also applied in each coding process of a four-level video. For pixels with their gray levels within the threshold band, their bi-level values can be modified according to the prediction of the adaptive context-based arithmetic encoding.

Table 1. The bit rates of the second bit planes of some testing video clips using an intuitive method and our method.

Video name	Intuitive method (bps)	Our method (bps)	Reduced
Akiyo	13030	11591	11.0%
Salesman	25082	21688	13.5%
Missbm	14523	12684	12.7%

Table 1 shows the bit rates of the second bit planes of some standard testing video clips using an intuitive method and our

method. In the intuitive method, the context for the intra or inter template of a pixel in context-based arithmetic coding is retrieved only from the current bit plane while in our method, the context is determined with taking into account its preceding bit planes as described above. Our method reduces the bit rates of the second bit planes by about 11-13%. The reduction will be more significant in videos with more levels.

Table 2. The Bit rate of each bit plane in four-level videos

Video Name	First bit plane (bps)	Second bit plane (bps)	Ratio
Akiyo	7148	11591	1.62
Salesman	14693	21688	1.48
Missbm	7336	12684	1.73

Table 2 shows the bit rates of each bit plane in a four-level video generated by our method. Surprisingly, the total bit rate of the second bit plane is about 1.4-1.7 times of that of the first bit plane. Intuitively, since the bit stream of the second bit plane (as in Fig. 2(k)) is the combination of the bit streams from mutually exclusive regions in two bi-level videos (as in Fig. 2(i) and Fig. 2(j)) and the total area is the same as that of the first bit plane, the bit rate of the second bit plane should be almost the same as that of the first bit plane. In order to investigate the problem, we counted bit rates from different regions of the two bi-level videos. We define the useful region as the region in the two bi-level videos that contributes bits to the second bit plane of the four-level video, e.g. the region in Fig. 2(i) marked by brighter pixels in Fig. 2(d) and the region in Fig. 2(j) marked by darker pixels in Fig. 2(d). We define the useless region as the region in the two bi-level videos that does not contribute bits to the second bit plane of the four-level video, e.g. the region in Fig. 2(i) marked by darker pixels in Fig. 2(d) and the region in Fig. 2(j) marked by brighter pixels in Fig. 2(d)). The bit rates of useful regions and useless regions in the bi-level videos (as in Fig. 2(i) and Fig. 2(j)) of the Akiyo video clip are shown in Table 3. The sum of the bit rates of the useful regions in the two bi-level videos is exactly the bit rate of the second bit plane of the four-level video. In this table, we find that the bit rates of the useful regions are much higher than that of the useless regions while the total bit rate (list at the third column of the table) of each bi-level video is similar to that of the first bit plane of the four-level video (see Table 1). The reason is that the values of the pixels in the useless region can be determined from the first bit and are equal to a same value (0 in Fig. 2(i) and 1 in Fig. 2(j)), while the useful region possesses higher complexity and accumulates most of the entropy of the image. Since the second bit plane collects the complexity of both of the bi-level videos, it consumes more bits.

Table 3. The bit rates of useful and useless regions in the two bi-level videos of Akiyo video clip

	Useful region	Useless region	Sum
Bi-level video with higher threshold	5791	1267	7058
Bi-level video with lower threshold	5800	1342	7142
Sum	11591	2609	14200

After careful study, we find that the bit rate of the second bit plane can be further reduced. As shown in Fig. 2, when we code the two bi-level image sequences of the second bit plane, we actually train the possibility table using both the useful region and useless region. While the useful region characterizes the features of the image content, the useless region dilutes it. Therefore the resultant possibility tables are not efficient. By only training the tables on the useful regions, we reduced the resultant bit rates by about 4-7% (see Table 4).

Table 4: The improvement of the bit rates of the second bit plane

Video name	On the whole image (bps)	On the useful region only (bps)	Reduced
Akiyo	11591	11012	5.0%
Salesman	21688	20696	4.6%
Missbm	12684	11771	7.2%

5. Multiple Level Video Coding

The coding of four-level video can be easily extended to the coding of multiple-level video. From the third bit plane of a multiple-level video, thresholds are chosen by equal subdivision between thresholds in the preceding bit planes. If the grayscale level of an image region cannot be further divided in some high level videos, the dividing stops. This situation can also be detected by the decoder and will not affect decoding. In addition, the context of an intra or inter template in a certain bit plane can also be determined with regard to its preceding bit planes. Finally, the coding of the ith bit plane ($i \geq 1$) is just the combination of the coding of 2^{i-1} bi-level video parts. The total computational cost of a 2^i level video is about i times of that of a bi-level video.

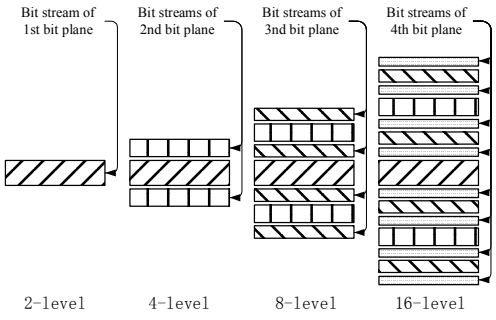

Figure 7: The bit streams of multi-level videos.

Fig. 7 shows the bit streams of each bit plane of multi-level videos. We represent bit streams of different bit planes with different patterns and order these bit streams from top to bottom just as the positions of the thresholds that are used to generate them. When the coded bits of a frame are output, they are output in independent bit streams. For example, the bits of each frame of an 8-level video consists of one bit stream for the first bit plane, two bit streams for the second bit plane and four bit streams for the third bit plane. Each bit stream corresponds to a bi-level video part that is converted from certain gray level region of the original grayscale video using a certain threshold. Bit streams in each bit plane do not affect each other. This results in a very good property

in error resilience. We can see that each higher level video contains all the bit stream of a lower level video. Each bit stream of a higher bit plane details the gray levels represented by a bit stream in its lower bit plane.

(a) (b)

Figure 8: Frames of an 8-level video without (a) and with (a) downsampling in the third bit plane.

In order to further reduce the bit rates of bit planes from the third bit plane, a downsampling method is introduced. An image is divided into 2×2 pixel blocks. If values of the 4 pixels in all bit planes that are higher than the current bit plane (at least the third bit plane) are the same, the average binary value in the current bit plane of the 4 pixels is regarded as the binary value of all the 4 pixels in the current bit plane. When pixels are coded in raster order, the binary values of all the other 3 pixels in the current bit plan are predicted but not coded except that of the bottom-right one. The average binary value of the four pixels in the current bit plane is calculated when the bottom-right one is coded. We need not spend bits to indicate these special blocks since they can be identified in the decoder by the same information on the higher bit planes. Inversely, the binary values of all the other 3 pixels in the current bit plane are not set until the bottom-right pixel is decoded. Fig. 8 shows the images without and with downsampling in the third bit plane of an 8-level video. The visual effects are almost the same, but the total bit rate is reduced from 66846 bps (a) to 54903 bps (b).

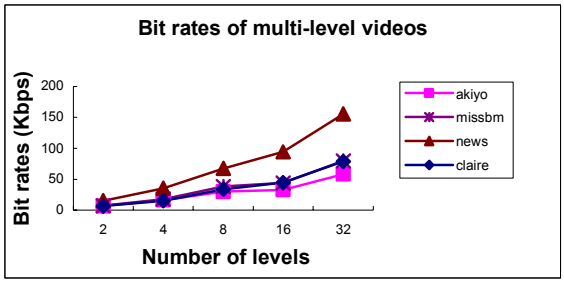

Figure 9: The bit rates of multi-level videos.

The bit rate distributions of multi-level videos with different grayscale levels are shown in Fig.9. The bit rate of a 2^{i+1} level video is about 2.4 times of that of a 2^i level video (where $i \geq 1$). If the number of levels of a multi-level video is greater than 4, its compression ratio is no longer competitive with that of DCT-based methods. In video communication applications, we usually use bi-level, tri-level and four-level videos to cover the 10-40 Kbps bandwidth range in 2.5G wireless networks. In some network conditions where bandwidths vary significantly, if the applications, e.g. security monitoring, require uninterrupted switches between low quality and high quality videos, we use multi-level videos with all grayscale levels, otherwise we switch to traditional full-color video.

6. Multi-level Video Transmission

In video streaming (video broadcast) applications, a full grayscale video can be encoded into a multi-level video in advance while in video communication applications, the captured video is encoded in real-time according to demands. For a given bandwidth, we can send corresponding levels of a video. If the bandwidth changes, the number of the video levels can be increased or decreased. Besides increasing or decreasing the number of video levels directly from 2^i to 2^{i+1} (where $i \geq 1$) or from 2^i to 2^{i-1} (where $i \geq 2$), we can also increase or decrease the number of video levels from j to j+1 (where $j \geq 2$) or j to j-1 (where $j \geq 3$) (see Fig. 10) since the bit stream corresponding to each threshold in a bit plane is independent and can be exported individually. This makes the transition of videos between different levels smoother and fits the video into the given bandwidth more precisely. As indicated in Fig. 10, when we need to increase the number of gray levels of a video, we usually add streams with their corresponding thresholds sorted from the principal threshold to the two sides. This is because gray levels near the principal threshold usually represent richer contents of the image.

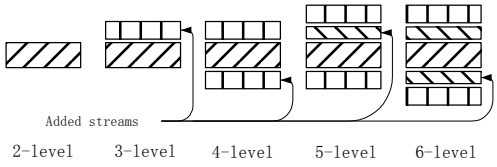

Figure 10: The gray levels of a multi-level video can be increased or decreased by one.

Another method that we can use to make a smoother transition of videos between different levels is fading in (when the number of video levels increases) or fading out (when the number of video levels decreases). This method depends on the full grayscale capability of the display. For example, when we have greater bandwidth and start switching a bi-level video to a tri-level video, instead of immediately displaying the tri-level video, we fade it into

(a) (b) (c) (d) (e)

Figure 11: The fading in effects in a switch from a bi-level video to a tri-level video.

the bi-level video. Since the first bit plane of the tri-level video is actually the original bi-level video, we still have the original bi-level video to complete the fading in operation when we are receiving a tri-level video. The effects are that the brighter areas of a bi-level video are unintentionally displayed in two levels of brightness in about 1-2 seconds (see Fig.11). When we have to switch a tri-level video to a bi-level video, the fading out operation is not so easy, since when we are receiving a bi-level video, we no longer have the tri-level video to complete the fading out operation. In some cases, a solution is that when the available bandwidth of a network decreases, we do not immediately switch to a bi-level video, but instead the sender sends a fading out signal to the receiver, which means to start fading out while continuing to send the tri-level video for 1-2 seconds. This gives the receiver the time and resources to complete the fading out operation before displaying the pure bi-level video.

7. CONCLUSIONS

We have described the whole procedure of the generation, coding and transmission of multi-level video. We call this kind of multi-level video portrait video because the coding is ordered from outlines to details, and the videos with lower levels look like a portrait. Much different from DCT-based coding method, which puts first priority on the average colors of a scene, portrait video puts first priority on the outline of a scene and then adds more details to it if more levels are involved. In some sense, portrait video always delivers the most important information of a scene for a given bandwidth. Portrait video is scalable because each video of a higher level always contains all the information of the video of a lower level and enhances the lower level videos.

Portrait video scheme possesses a number of features. We select suitable thresholds for ordinary scenes based on Ostu's Between-class variance method, and allow users to fine tune in video communication scenarios with the assistance of our empirical method. These methods ensure that more important information of a scene is included in lower bit planes, which will be transmitted with high priority. By analyzing the composition of each bit plane of a multi-level video, we convert the compression of noisy bit planes to the compression of several regular bi-level video parts, thereby greatly improving the compression ratio. This also results in multiple independent bit streams, which facilitate the switch between videos with different levels and error corrections. Moreover, a series of methods including improved possibility table training, downsampling, fading in and fading out are designed to ensure efficient compression and a smooth transition of portrait videos.

While the present method can also be applied to general bit plane coding in other image processing and video compression technologies, it is particularly promising in mobile video conferencing on 2.5G wireless networks. This is because the bandwidths of 2-4 level portrait videos fit into the bandwidth range of 20-40 Kbps that GPRS and CDMA 1X can stably provide, and the cheap computational costs of 2-4 level videos are affordable by mobile devices.

We have applied portrait video technology to Microsoft Portrait, a research prototype for mobile video conferencing. With portrait video technology, we are the first to enable mobile video conferencing on an iPAQ Pocket PC with an HP Jornada pocket camera. Microsoft Portrait can be freely downloaded from http://research.microsoft.com/~jiangli/portrait/.

8. REFERENCES

[1] 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.

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

[3] ITU-T Recommendation H.261 Video codec for audiovisual services at p x 64 kbit/s, 03/93.

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

[5] N. Ostu, A threshold selection method from gray-level histogram, IEEE Trans. Systems Man Cybernet. SMC-8, 1978, 62-66.

[6] T. Pun, A new method for gray-level picture thresholding using the entropy of the histogram, Signal Process. 2, 1980, 223-237.

[7] J. N. Kapur, P. K. Sahoo, and A. K. C. Wong, A new method for gray-level picture thresholding using the entropy of the histogram, Comput. Vision Graphics Image Process. 29, 1985, 273-285.

[8] J. Kittler and J. Illingworth, Minimum error thresholding, Pattern Recognit. 19, 1986, 41-47.

[9] W. H. Tsai, "Moment-preserving thresholding: a new approach," Computer Vision, Graphics, and Image Processing, Vol. 29, 1985, pp. 377-393.

[10] R. L. Kirby and A. Rosenfeld, A note on the use of (gray-level, local average gray level) space as an aid in the thresholding selection, IEEE Trans. Systems Man Cybernet. SMC-9, 1979, 860-864.

[11] A. D. Brink. Thresholding of digital images using two-dimensional entropies. Pattern Recognition 25. 1992, 803-808.

[12] N. Papamarkos and A. Atsalakis, "Gray-level reduction using local spatial features," Comput.Vision Image Understanding, vol. CVIU-78, 2000, pp.336–350.

[13] P. K. Sahoo, S. Soltani, and A.K.C. Wong, "A survey of thresholding techniques", Computer Vision Graphics Image Processing. 41, 1988, 233-260.

[14] P. Liao, T. Chen and P. Chuang, "A Fast Algorithm for Multilevel Thresholding", Journal of Information Science and Engineering 17, 2001, 713-727.