Personal Photo Album Compression and Management

Ruobing Zou, Oscar C. Au, Guyue Zhou, Wei Dai, Wei Hu and Pengfei Wan
Department of Electronic and Computer Engineering
Hong Kong University of Science and Technology
Hong Kong SAR, China
line 4: {zouruobing, eeau, zhouguyue, weidai, huwei, leoman}@ust.hk

***Abstract*—The advance in multimedia technologies have resulted in an explosive growth of pictures in personal computers and in cloud. Typically many pictures taken in the same occasion are similar. The cost to store and transmit them can be very significant. Thus it is important to find an efficient method to store these pictures. This paper proposed a compression scheme for similar images. Our approach is to arrange all the similar images into tree structure then apply video coding technique along each branch. To maximize the inter-image correlation between adjacent photos, we consider the minimum spanning tree (MST) subjecting to a maximum depth limit to ensure fast access to all images. This structure is encoded by the latest video coding technique High Efficiency Video Coding (HEVC), which is reported to has advantage in high definition video/image compression. It also supports deleting, adding and modifying images. Experiments show that the proposed method saved 75% space comparing to JPEG format.**

I. INTRODUCTION

As the fast development of multimedia technologies and the prevailing of digital cameras, pictures taken in daily life are of higher resolution and bigger size. They gradually eat up the hardware storage space. Usually, images exported from cameras have been compressed by traditional techniques like JPEG. Nevertheless, further compression can still be achieved by eliminating the redundancy between these JPG photos. Such measures including using graph theory to build a minimal-cost prediction structure are proposed and refined in [1, 2, 3]. Likewise, [4] developed a Generic Quad-Tree (GQT) approach to store biological similar images, and [5] studied image sorting in frequency domain before compression. These previous explorations laid the foundations for our work.

However, personal photo album has some unique characteristics that make it difficult to compress. Unlike the linear video sequence which has strong inter-frame correlations, the similarity between two chronically adjacent pictures is unknown. Moreover, differing from medical image sets where the images are largely similar, personal photos involve frequent scene changes and larger motion offset. It hence may require special processes such as similarity measure and larger motion estimation search range.

II. PRE-PROCESSING AND SIMILARITY MEASURE

To better understand the attributes of albums, it is necessary to define "similarity" in the first place. Before further study, pre-processing is needed by first resize the images to a size between 250-500 pixels in width to reduce complexity. Also, as our scheme uses the YUV color space, the Y-component is normalized to a value in the range of (0,1), in order to reduce the intensity scale difference caused by changing lighting and ISO.

Then the image registration techniques [6] help address this problem. Given a nonempty set S of N photos in the same size, the similarity between two images $i, i' \in S$ is measured by $SSD(i,i')$ the sum of the squared differences (SSD) by block-based motion estimation. The image to be encode is divided into NxN blocks. For each block B_k in i', it finds a best matching block B'_{match} in the reference image i that minimizes the SSD. Then the total SSD for all K blocks in image i' are summed up:

$$SSD(i,i') = \sum_{k=1}^{K} \sum_{x=1,y=1}^{N,N} (B_k(x,y) - B'_{match}(x,y))^2 \quad (1)$$

As we can see, $SSD(i,i')$ also indicates the prediction cost: a small value indicates i similar to i', thus i is suitable to be a reference. When it comes to "similar images", we have:

Definition 1. *Let $\theta \in [0,1]$ be a fixed image similarity threshold. Two images $i \in S$ and $i' \in S$ are similar if $D(i, i') \le \theta$; and they are dissimilar otherwise.*

Then by extending this concept to image sets, we can measure the similarity of an image set. It is calculated as the percentage of similar image pairs in all possible image pairs. More formally, let $S_S(i,i')$ be the set of similar image pairs in S, and $S_A(i,i')$ be the set of all pairs. Also, let $|S_S(i,i')|$ (or $|S_A(i,i')|$) denotes the cardinal of $S_S(i,i')$ (or $S_A(i,i')$). The similarity of set S is $sim(S)$, and likewise we can define "similar image set":

$$sim(S) = \frac{|S_S(i,i')|}{|S_A(i,i')|} \quad (2)$$

Definition 2. *Let* $\sigma \in [0,1]$ *be a fixed threshold for percentage. If* $sim(S) \geq \sigma$, *then this image set is considered to be similar; otherwise it is dissimilar .*

Above determines whether an album is a similar image set. A similar album S is eligible to fit into the graph framework in next section (Section III). However, it is inefficient to directly input a dissimilar album into the framework which exploits inter-image redundancy, because this album may contain a large portion of discrete images that has little correlation with others. Hence their contribution is relatively small to improve compression efficiency. To do this will also take a substantial amount of time and resource. For this reason, we extract the similar images from a dissimilar set, and only model this set $S' \subset S$, while the dissimilar images are left intact until the HEVC encoding stage.

III. IMAGE SET MODELING

A. Graph Theoretical Framework

We use a weighted directed graph $G = <V, A>$ to model an image set S (for similar images set) or S' (for dissimilar set) containing N image. Images are vertices $V = \{v_1, v_2 \ldots v_N\}$, each connected to all other $N-1$ vertices. Arcs $A = \{(v_i, v_j) | i=1,2...N, j=1,2...N\}$ represent the predictive relationships: $arc(v_i, v_j)$ $(i \neq j)$ denotes predicting vertex v_j using the reference image v_i by inter prediction. The arc has a weigh $w(v_i, v_j) = SSD(v_i, v_j)$. Similarly, the loop $arc(v_i, v_j)$ $(i=j)$ denotes exploiting the spatial redundancy within a single image by intra prediction. Its weight $w(v_i, v_j)$ is calculated as the intra prediction for 16x16 macroblock in H.264/AVC.

B. The Optimal Prediction Structure (OPS)

Based on the above graph model, we want to find an optimal prediction structure to determine how images are predicted. A typical prediction structure in video compression is a linear sequence of IPP…P structure, but for photos, it would incur longer time delay to access a picture in the rare of a sequence. Therefore, a tree structure is introduced, which includes one or several trees with minimized total prediction cost, whose roots are intra-predicted and every image other than the root are inter predicted.

Theoretically, it equals to finding a subgraph to minimize total weight. Suppose the OPS contains k trees, whose roots are $R=\{r_1, r_2, \ldots r_k\}$, we use $T(r_m)$ to denote a tree whose root is r_m . And the set of trees in the OPS is denoted as $T(R) = \{T(r_m) | m = 1,2...k\}$. We also to denote the vertexes in a tree $T(r_m)$ as $V'(T(r_m))$. The problem can be described as:

$$\arg\min_{T(R)\subset G, R\subset V} \sum_{r_m \in R, m=1}^{m=k} \left(w(v_{r_m}, v_{r_m}) + \sum_{(v_i, v_j)\in T(r_m)} w(v_i, v_j) \right) \quad (3)$$

$$\text{subject to } V'(T(r_1)) \cup V'(T(r_2)) \cup \ldots \cup V'(T(r_k)) = V,$$

$$\text{and } V'(T(r_1)) \cap V'(T(r_2)) \cap \ldots \cap V'(T(r_k)) = \phi$$

We add the constraints to make sure all vertexes are included and no shared vertexes are allowed between trees.

To solve this problem, we change G to a new graph $G_e = (V_e, A_e)$ by adding a virtual vertex to this graph, and replacing each loops respectively with an arc (v_v, v_i) at weight $w(v_v, v_i) = w(v_i, v_i)$. The problem can be mapped to finding a spanning tree ST in the new graph G_e that minimize:

$$\arg\min_{ST\subset G_e} \sum_{(v_i, v_j)\in ST} w(v_i, v_j) \quad (4)$$

A minimum spanning tree (MST) is constructed from G_e by applying the Chu/Liu and Edmond algorithm, assuming the virtual node as root. Having found the MST, the OPS is found by deleting V_v and its outgoing arcs. The whole process is illustrated in figure 1.

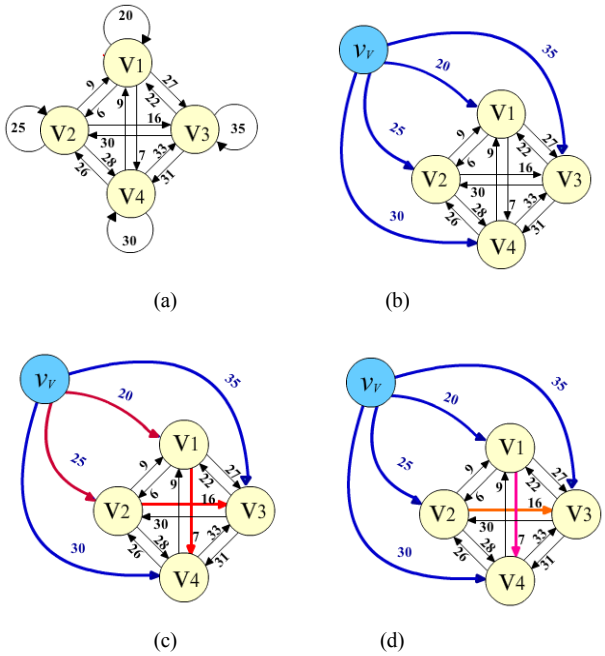

Figure 1. Example of a graph model and its OPS. *((a) A graph model of 4 images. (b)The extended graph* G_e *. (c)The MST of* G_e*. (d) The final OPS.)*

C. Tree Depth Constraint

Another important problem is to guarantee users of a fast access to read or search any image. However, taking the leaves for example, then the deeper a tree, the longer time it will take to reach them, because more precedent images will need to be decoded. Moreover, it occupies extra memory space to store the decoded precedents into the reference images' buffer. Therefore, we impose a maximum depth constraint d to all trees in an OPS G . The value of d indicates how long users are willing to wait for decoding an image. This idea is implemented by re-connecting the vertices as below.

Given a tree, we first build an acyclic graph T from it. The vertices are renumbered in a topological order, and let the depth of each vertex v_i be $dep(i)$, T is constructed by adding a new arc (v_i, v_j) between every pair of vertices v_i and v_j that satisfies $dep(i) < dep(j)$. The weights of these arcs are still $w(v_i, v_j)$ as in section III. Then the task becomes finding a spanning tree $\tilde{T}$ which subjects to the depth constraint $dep(\tilde{T}) < d$ while having the minimum cost of $\sum_{\tilde{T}} w_{\tilde{T}}$.

The derivation of an optimal $\tilde{T}$ is by checking the summed weight of sub-trees bottom-up, from leaves to root. We describe the state of a set of vertices $v = \{v_1, v_2 \ldots v_n\}$ as $s = \sum_{i=1}^{n} 2^{v_i}$. Notice that $v = \varnothing$ leads to s = 0. Then let the

function $f_{i,j,s}$ denote the minimum total weight of a sub-tree rooted at vertex v_i, whose depth is j and vertices' state is s. $\tilde{T}$ can be found by solving (4) throughout T in an inverse topological order, that is, starting from the leaves, we calculate:

$$f_{i,j,s} = \min\{f_{l_1,j_1,s_1} + f_{l_2,j_2,s_2} + \ldots + f_{l_m,j_m,s_m} + \sum_{p=1}^{m} w(i,l_p) \quad (5)$$

$$\text{Subject to: } \max\{j_1, j_2 \ldots j_m\} = j-1, \ \sum_{p=1}^{m} s_p + 2^{i-1} = s.$$

Here $L=\{l_1, l_2,\ldots, l_m\}$ is a subset of vertex v_i's children, intuitively meaning that children in L are selected in the current iteration to build a sub-tree while the other children are not. The latter element $w(v_i, l_p)$ is arc(v_i, l_p)'s weight.

This tree reconstruction algorithm is performed on all trees whose depth exceeds the maximum threshold, so that the depth of the whole forest will not be larger than d.

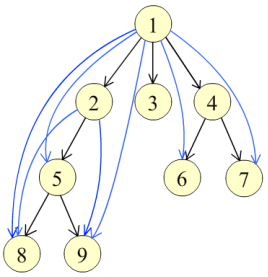

Figure 2. Graph T for depth constraint.

IV. Compression Scheme

Since most of the digital pictures are of high definition, we adopt HEVC as coding technique. A group of picture (GOP) is proposed in our work to organize the pre-acquired minimum spanning forest into a consecutive image sequence. The key idea is to compress every tree in the forest as an independent GOP, starting from the root being encoded as an intra frame, followed by the rest of its successors coded as inter frames predicted by its reconstructed parent. The tree is scanned in a depth-first search (DFS) order. The GOPs of all the trees forms an image sequence, as illustrated in figure 3.

The applicable video coding standard is not limited to HEVC, also applicable are the H.264/AVC, MPEG-4 etc.

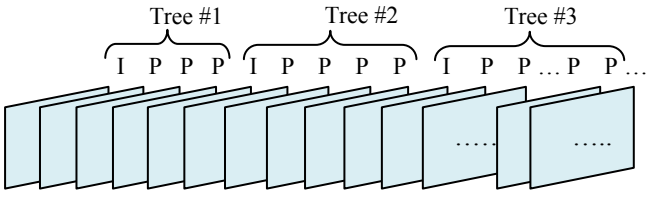

Figure 3. Image sequence.

V. Album Management

An extension to photo album compression is the file management. In this section, we describe three elementary operations involving image inserting, deleting and modification. [4]

A. Insertion of New Images

One intuitive understanding of this problem is that adding new images into an existing album essentially equals to inserting new vertices at once into graph G, so one solution would be to compute the OPS again. But provided that the new vertices end up parents of existing vertices, it will be quite inefficient because it may require rerunning HEVC for the whole updated branch. A simpler way shall be directly adding the new images to G' one by one as leaves instead. Suppose images $\{I_1\ I_2 \ldots I_a\}$ are added, and the roots in G' are $\{r_1\ r_2 \ldots\ r_b\}$, the pseudo-code is:

for new image $I_i \leftarrow I_1$ to I_a
  for root $r_j \leftarrow r_1$ to r_b
    Compute $w(r_j, I_i)$ that is the cost of building an arc
  Find the $r_{j'}$ with the minimum $w(r_j, I_i)$, suppose $r_{j'}$ belongs to tree T
  for vertex $v_k \leftarrow v_1$ to v_c in T
    if v_k is a leaf && $dep(T) = d$
    then skip vertex v_k
    else compute $w(v_k, I_i)$ that is a arc's building cost
  Find the $v_{k'}$ with the minimum $w(v_k, I_i)$
  Assign I_i to be a child of $v_{k'}$ and build an arc $(v_{k'}, I_i)$

After finishing the above steps to update the forest, we compress the leaves i.e. new images using HEVC with their parent as prediction references; and the new encoding results are written into the existing image sequence bitstream.

B. Deletion of Images

As for the deletion, the main concern is to avoid changing the internal predictive relationships inside a tree, which may otherwise lead to rerunning HEVC. So the basic strategy of deleting an image from a tree is to mark this vertex as "deleted" without actually removing its information from the encoded sequence, unless the vertex is a leaf or its successors have all been marked as "deleted". In the latter case, we can regard it as a leaf of a "trimmed tree". Suppose images $\{I_1\ I_2 \ldots I_a\}$ are to be deleted, the basic steps are:

for image $I_i \leftarrow I_1$ to I_a
  if I_i is not a leaf in its tree or the "trimmed tree"
    Mark I_i as "deleted"
  else delete all information from both the tree and the encoded sequence

C. Modification of Existing Images

Image modification is substantially a combination of deletion and insertion. The basic steps of manipulating modified images $\{I_1\ I_2 \ldots I_a\}$ are: first, delete the vertex of original image I_i from the tree; second, inserting the modified image I_i' to the forest and compress I_i' into the image sequence.

VI. Experimental Results

Experiments are conducted on two photo albums of 150 personal pictures each. The pictures in album 1 are taken by a Digital Single Lens Reflex (DSLR) camera NIKON D90 at

the resolution of 4288×2848; and that of album 2 are by a digital still camera, at the resolution of 2592×1944. The albums both contain pictures from various events in daily life, like trip view, party scene, portraits and architectures etc. The encoding/decoding software is HEVC Test Model HM 7.0.

For parameters chosen, the similarity measure thresholds are $\theta = 0.8$, $\sigma = 90\%$. The block size for arc weight estimation is 16×16, and exhaustive search is used.

1) Improvement percentage: To give an overall view, we compare the Information Bitrate(IB), or the average bits per pixel (bpp), defined as IB=TotalBits/TotalNumberOfPixels [7]. We also compare the Improvement Percentage of our scheme over JPG: IP=1−TotalBits (IsOPS)/TotalBits(IsJPG) [1]. The similarity of the album are *sim*(A1) = 0.85 indicating that this is a dissimilar album; and *sim*(A2) = 0.95 meaning it's a similar album with regard to the value of θ and σ. Table I show the results when the quantization parameter (Qp) in HM is fixed at 22, which is relatively a small value that preserves much of a picture's quality. As the results show, our method reduced 75% the file size as in JPG format.

TABLE I. IMPROVEMENT PERCENTAGE

Image Set	*IB(bpp)*		*IP*
	JPG	***Proposed***	***Prop. over JPG***
Album 1	2.82	0.66	76.3%
Album 2	1.95	0.48	75.4%

2) PSNR comparison: we compare the performance of (*a*) our OPS+HEVC method and (*b*) directly using HEVC to compress the unsorted albums whose pictures are in a chronicle order. For (*b*), it neither does motion estimation to get weight nor constructs an OPS, and uses a picture's temporally adjacent previous picture as a predictive reference. The results in figure 4 are generated under Qp values {22, 27, 32, 37}, in terms of PSNR and bpp. The Y-channel PSNR achieved an average gain of 0.35dB and 2.1dB respectively.

3) Complexity: since it is possible to directly apply HEVC to compress the JPG images as in experiment 2), we summarize the complexity increased due to similarity comparison, graph constructing, OPS calculating and depth constrain. Under the same test condition, the time increased by our method is 9.2% of HEVC. In other words, the increase in complexity is within an acceptable range. Additionally, after examining the time profiling, it is found that more than 85% of the extra time complexity concentrates on arc weight estimation, suggesting further space for optimization.

VII. CONCLUSION

The graph-based compression method presented in this paper provides an efficient solution to store and manage personal digital photos. By modeling the predictive structure as a minimum spanning forest, it takes into account both the spatial correlation of images and the temporal correlations between images to reduce redundancy. Together with the depth constraint algorithm, it allows fast access to images in the structure. The proposed scheme also guarantees users of the basic operations to insert new image, delete images, and modify images. Our future work would be focusing on reducing complexity in calculation arc weights, while keeping the accuracy of weight estimation.

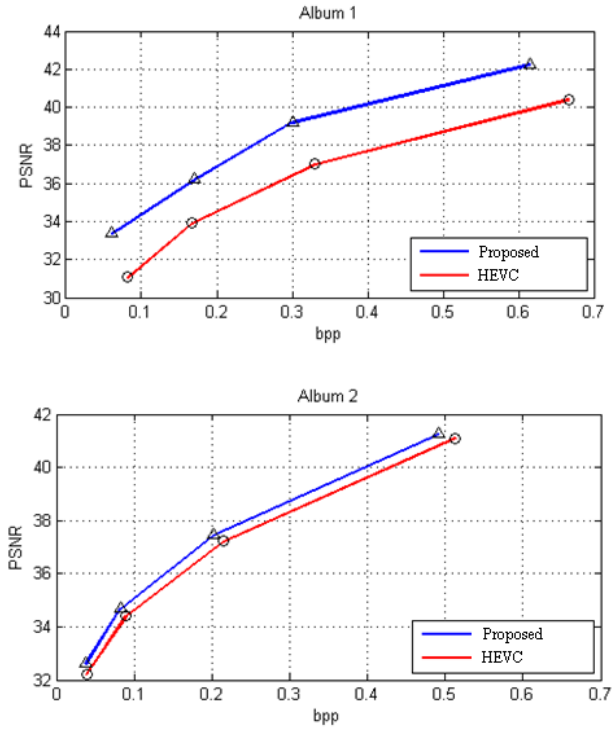

Figure 4. Compression results

ACKNOWLEDGMENT

This work has been supported in part by the Research Grants Council (RGC) of the Hong Kong Special Administrative Region, China.

REFERENCES

[1] C. Nielsen, X. Li, "MST for lossy compression of image sets," Data Compression Conference (DCC'06), pp.463, 2006.

[2] C. Chen, K. Chung, H. Lu, and G.Y. Tang, "Image set compression through minimal-cost prediction structures," Int. Conf. on Image Processing (ICIP'04), pp.1289-1292, 2004.

[3] B. Gergel, H. Cheng, C. Nielsen, and X. Li, "A Unified Framework for Image Set Compression," Int. Conf. on Image Proc., Comp. Vis., and Pat. Rec. (IPCV'06), pp.417-423, 2006.

[4] G. Jomier, M. Manouvrier, and M. Rukoz. "Storage and management of similar images." Journal of the Brazilian Computer Society, Vol. 6(3), pp. 13-25, 2000.

[5] Y. Lu, T.T. Wong, and P.A. Heng, "Digital photo similarity analysis in frequency domain and photo album compression," Int. Conf. on Mob. and Ubi. Mul., Vol. 83, pp. 237-244, 2004.

[6] L. G. Brown, "A survey of image registration techniques," ACM Comput. Surv. Vol. 24(4), pp 325-376, 1992.

[7] C. H. Yeung, O. C. Au, K. T. Tang, Z. D. Yu, E. Luo, Y. Wu, and S.F. Tu, "Compressing similar image sets using low frequency template," Int. Conf. on Mult. and Expo.(ICME'11) , 2011, pp.1-6.