Compression of Polynomial Texture Maps

Giovanni Motta, Marcelo J. Weinberger
Information Theory Research Group
HP Laboratories Palo Alto
HPL-2000-143 (R.2)
April 24th , 2001*

image compression, texture mapping, 3D graphics, JPEG, JPEG-LS, polynomial texture maps

Polynomial Texture Maps (PTM's) are representations of stationary 3D objects which provide enhanced image quality and improved perception of the object surface details at the cost of increased storage requirements. In this paper, we investigate several compression schemes for PTM's, that either preserve every feature of the original texture map (lossless compression), or allow for a small reconstruction error (near-lossless and lossy compression).

1. Introduction

Polynomial Texture Maps (PTM's) were introduced by Malzbender et al. in [1] as an extension to conventional texture mappings. PTM's represent stationary 3D objects while providing enhanced image quality and improved perception of the object surface details. This representation is derived from a series of pictures of the same object taken with a conventional, stationary, digital camera under slightly different conditions (light direction, focus plane, illumination). Since both the camera and the object are stationary, the set of pixels in a common location carries information on the object surface and its behavior under changes in illumination. Combining the information of the pixels in corresponding locations (x, y) in the sequence of images, intensity dependencies on the light direction (l_x, l_y) are modeled with second-order bi-quadratic polynomials. For example, for the surface luminance L,

$$L(x, y; l_x, l_y) = a_0(x, y)\, l_x^2 + a_1(x, y)\, l_y^2 + a_2(x, y)\, l_x\, l_y + a_3(x, y)\, l_x + a_4(x, y)\, l_y + a_5(x, y) .$$

One possible approach is to store for each pixel (x, y) the six polynomial coefficients $a_0,\dots,a_5$ (i.e., its luminance model), in addition to its color intensity values. When reconstructing images under varying illuminations, the color intensity values are scaled by luminance.

At the cost of increased storage requirements, PTM's have the advantage that more information is available for each pixel and this information can be used to enhance the rendering and improve the visual interpretation of the represented 3D object. Applications in several fields such as archeology, forensics, medicine, quality control, etc., can take advantage of this representation. In this paper, we report on an investigation aimed at designing methods for reducing the storage requirements of PTM's while either preserving every feature of the original texture map (lossless compression) or allowing for a small reconstruction error (near-lossless and lossy compression). When a reconstruction error is allowed, it is important that the most interesting features of the PTM representation are carefully preserved.

Two different PTM formats were addressed: LRGB and RGB. In LRGB format, each pixel is represented by 9 values, namely the six polynomial coefficients a_i^I, $0 \le i < 6$, that model its luminance as a function of the light direction, and the three Red, Green, and Blue (R,G,B) color components. The richer RGB format uses 18 values that represent three different intensity models, one for each color component: a_i^R, a_i^G, a_i^B, $0 \le i < 6$. The polynomial coefficients are stored after appropriate bias cancellation and scaling, rounded to the nearest integer so that they always assume values in the alphabet $\{0, \dots, 255\}$. Original coefficients can be reconstructed (with some loss due to the rounding) as:

$$a_{final} = (a_{raw} - bias) * scale$$

More details on these and other PTM formats can be found in [2]. (Following [2], we call "textel" the set of coefficients that represents a pixel.)

Compression Modalities

Three main compression modalities were investigated, each aimed at a different application of PTM's:

1. **Lossless compression**: useful for archiving the maps. When a PTM is compressed in lossless mode, it is always possible to reconstruct the uncompressed original without any error. Because of the perfect reconstruction requirement, lossless coding achieves the highest bit rate among the three methods and the compressed maps are sometimes still too large for low bandwidth applications.
2. **Near-lossless compression**: the compression algorithm is allowed to introduce some coding error so that the absolute difference between each reconstructed pixel and its original value is bounded by a fixed quantity that is predefined in the coding process. Typical values for the maximum error range from 1 to 5, depending on the image and on the targeted bit-rate. Near-lossless compression is useful when a lower bit rate is required but the image quality cannot be compromised in an unpredictable manner.
3. **Lossy compression**: this modality achieves the lowest bit rate among the three methods. The compressed image can still be used for most applications and is in general virtually free from visible artifacts, but there is no guarantee that the absolute value of the error on each pixel is limited in a uniform and predictable way. While we distinguish between near-lossless and lossy compression based on the way the error is introduced, near-lossless is of course a special case of lossy compression.

2. Description of the Algorithms

Display of each coefficient plane as an image shows that planes look very similar and each plane preserves most of the characteristics of the object that is represented (see Fig.1 and Fig.2). Accordingly, we follow an approach common in digital compression of multi-spectral images. Multi-spectral images consist of several "bands," each representing a slightly different view of the same scene. RGB color images are an example of multi-spectral images, in which the bands are the color components red, green, and blue. A common approach to the compression of this kind of images consists in the determination of a decorrelation transform that removes the redundant information common to more than one plane and makes the planes "simpler" and ready to be compressed individually.

Ideally, in a sequential scheme, the probabilistic model used to compress a plane ought to be based on all previously encoded information, including neighboring pixels located in reference planes, as well as causal neighbors in the same plane. However, a two step approach with clear computational advantages consists of:

a) Inter-plane prediction: given a reference plane, it "guesses" the value of the pixels in the plane being encoded; and
b) Intra-plane probabilistic modeling applied to inter-plane prediction residuals: builds a probability model for the residuals by using the values of its (previously encoded) neighbors.

Similar considerations apply to non-sequential schemes (e.g. JPEG compression).

The first step is a preprocessing that applies to all compression modalities. Computing and executing the decorrelation transform first and compressing its result later, plane by plane, also allows for the use of off-the-shelf general purpose image compression algorithms for the compression of the individual planes. In general (see, e.g., compression of RGB images [7]), this two-step approach comes close to the compression achieved by global modeling approaches, and is thus adopted for PTM's.

The decorrelation is achieved by deciding for each plane its encoding mode (inter or intra), determining which reference plane must be used for the prediction (if the mode is inter), and by ordering the planes in a sequence that is compatible with a causal decoding. Inter-plane prediction requires the reference plane to be known to the decoder before the inter coded plane it predicts. To avoid error propagation, when in lossy modalities, the reference plane used in the prediction must not be the original one but, rather, its lossy decoded version. While the reference plane can be, in turn, inter or intra coded, at least one plane must be intra coded. The (total) ordering must be compatible with the partial ordering determined by the modes and by the reference planes. Coefficient planes are encoded and presented to the decoder according to this order, so that the decoder, before decoding an inter plane, has complete knowledge of its reference plane.

Cost Matrix and Optimal Ordering

Unlike in most compression algorithms involving inter-plane decorrelation, for which the relations between the planes, motivated by physical reasons, are known a priori (e.g., RGB image coding), such prior knowledge is not available in our framework. Instead, coding mode decision and coding order are determined on an image-by-image base, by an algorithm that builds a cost matrix and minimizes the total cost of the compression. The algorithm uses a fictitious "plane 0," in which every pixel value equals 0, for the purpose of building the cost matrix. Predicting a plane i from plane 0 is equivalent to compressing the plane i in intra mode. Thus, the mode decision is implicit in the cost minimization algorithm. The entry $cost(i, j)$ in the cost matrix represents the cost (say, in bits) of predicting plane i by using plane j as a reference. To speed up the matrix construction it is assumed that $cost(i, j) = cost(j, i)$, so that only half of the matrix must be computed. We generated matrices using various cost functions, such the plain entropy of the prediction error, prediction error entropy after application of the Median Edge Detector described in [3], and number of bits generated by an actual compression scheme with the chosen lossless or lossy encoding. Each cost function represents a trade-off between the compression achieved and the complexity of computing it.

The decorrelation sequence that minimizes the total cost is determined by interpreting the cost matrix as an adjacency matrix of a complete graph, and by finding its minimum cost spanning tree. Plane 0 will be a node of the spanning tree, and we can always assume the tree to be rooted at 0. Any topological ordering of the spanning tree rooted at 0 is clearly compatible with a sequential encoding and decoding (a similar approach is proposed in [8] for multispectral images). Figure 3 depicts a cost matrix, a minimum spanning tree, and a compatible plane ordering for the LRGB PTM image "trilobite.ptm".

Transforms

Inter-plane prediction is based on a simple difference between the plane being predicted and a reference plane. In order to achieve a more effective prediction, one or more transformations, selected in a small set, can be applied to the reference plane before it is used. Two transformations were determined to be effective in improving the prediction between planes:

a) Plane inversion, computed by complementing the value of each pixel; and
b) Motion Compensated prediction, a translation of the reference plane.

Plane inversion is motivated by the observation that some of the planes look like "negatives" due to the way the coefficients are generated. Complementing the value of each pixel in those planes, allows more choices for the possible decorrelation.

Motion Compensation (MC) is a technique widely used in video compression and it is well-motivated in this particular framework, since PTM images are generated by combining pictures of an object taken while changing the illumination conditions. Thus, the changes in the direction of the light result in small "movements" of the object's edges. It was found that MC of the order of one pixel is enough to achieve the highest decorrelation. Block-based motion compensation does not appear to be necessary since, while reducing the energy of the prediction error, it introduces blocking artifacts that may compromise the performance of the compression of the prediction error. Motion compensation is performed in two steps: first, we search for the integer displacement of the reference plane that achieves lowest prediction error; then, we refine the displacement of ±½ pixel along each coordinate. Half pixel refinement is performed by interpolating the reference plane and has the advantage of allowing smaller displacements while limiting the complexity of a full search.

After each plane has been decorrelated, a standard image compression algorithm can be applied to the planes (if intra coded) or to their prediction error (if inter coded). For our experiments we used JPEG-LS [3] for lossless and near-lossless compression, and JPEG [4] for lossy compression. Higher compression ratios in lossy mode, at the cost of increased computational complexity, can be obtained with the upcoming JPEG2000 standard.

Modulo Reduction

When subtracting a plane from another, the alphabet size for the prediction errors doubles (see [3] for a discussion on this topic); if the coefficients range between 0 and 255, a difference between coefficients can assume any value in the range [–255, +255]. In lossless compression, the reference value is known to the decoder, so it is possible to use modular arithmetic (or other similar mappings) to bring the alphabet of the prediction error to its original size (this technique also produces a slight improvement in the compression ratio). However, when an error is added in the encoding of the prediction error, modular arithmetic may cause overflows that are visible in the reconstructed image. Overflow errors appear on a limited number of pixels and are corrected by sending additional side information to the decoder. After the encoding, the reconstructed plane is compared to the original and if there is any overflow, the position of the corrupted pixel and its original value are described to the decoder. In near-lossless compression, overflows are detected by comparing the reconstructed plane with the original; a reconstructed pixel value that differs from the original by more than the predefined error bound, is an overflow. With JPEG, errors can be potentially large and their detection is less trivial. However, we are interested only in the correction of differences that are noticeable in the reconstructed image, so that comparing the error with a fixed threshold was found to be a good alternative.

3. Results

In assessing the compression results, it is important to assess the artifacts present after possible enhancements of the PTM images. In particular, the so-called "specular enhancement" algorithm (see [1]), which behaves as a strong contrast enhancement operator, is critical. Because of its behavior, compression artifacts are usually more noticeable after application of this algorithm. Since PTM images can be used in different settings and for different purposes, it is not possible to determine a single objective quality measure that captures the effects of the error introduced by lossy compression. Thus, visual assessment was the main tool to determine whether the final quality was acceptable.

Table I shows that the average compression ratio achieved in lossless mode on the set of LRGB and RGB images that we used for the tests, was 34.98 bits per textel (bpt). This result approaches the one obtained by a (lossy) vector quantization (VQ) scheme previously proposed to encode PTM images. Notice that while the palette indices in the VQ scheme are uncoded, further compression would require a hugely complex reordering of the color palette.

A 30% additional gain can be obtained by using near-lossless compression with a ±1 error bound (22.22 bpt on the average). By increasing the near-lossless error, higher compression ratios are achieved. It should be noted, however, that the use of near-lossless JPEG-LS with large error bounds (larger than 4) and with images that have large uniform areas, can produce a characteristic striping pattern. A modification of JPEG-LS to limit these errors is under investigation.

Higher compression ratios are achieved in lossy mode. When using JPEG compression on the decorrelated planes, a fidelity parameter between 75 (default) and 45, yields average ratios ranging between 7.81 and 4.44 bpt. Blocking artifacts (typical of low bit-rate JPEG encoding) are present in some images that have large smooth areas and are thus highly compressible (around 3 bpt). Those artifacts disappear with a quality factor of 55 or more (corresponding to 5.14 bpt or better). By embedding the upcoming JPEG2000 standard, a further reduction of the bit-rate might be possible while preserving the final quality.

Clearly, the RGB images in Table I show a significantly higher compression than the LRGB images (as the original depth is 18 bytes, as opposed to 9). This is explained by a higher impact of decorrelation, as the use of luminance in the LRGB representation already implies a form of decorrelation.

While our experiments show that some pairs of planes are strongly correlated across most images, the transformation that achieves the highest possible decorrelation for each image does not appear to be a single one. Table II shows the effect of using a fixed decorrelation for every image. It also shows a comparison between the compression obtained with decorrelation via the minimum spanning tree on the cost matrix, and the compression achieved by encoding every plane in intra mode (no inter-plane prediction).

4. Issues for Further Investigation

The following is a partial list of subjects that warrant further investigation:

- *Encoding of inter-coded and intra-coded planes with different quality factors*
  The loss introduced in inter-coded and intra-coded planes may affect differently the final quality of the rendering. Moreover, errors in the coefficients may have different visual effects. The current implementation of the algorithm uses identical qualities for all the planes.
- *Implementation of other transforms*
  Physical considerations may provide insight into preferred decorrelation schemes. These may include multiplicative (rather than additive) factors, logarithmic scales, etc.
- *Quality measures*
  The determination of an objective quality measure, including assessment of the usual MSE metric and study of the effect of artifacts in near-lossless JPEG-LS and JPEG, may lead to variants in the compression schemes.
- *Inclusion of JPEG2000 image compression*
  The bit-rates at which JPEG2000 may justify its additional complexity should be determined, as near-lossless compression tends to be effective at high bit-rates, whereas JPEG's performance in the medium range is quite reasonable.
- *Various minor improvements*
  Includes removal of JPEG-LS/JPEG header from the compressed data, and more compact encoding of side information.

Acknowledgement

Thanks to Tom Malzbender, Dan Gelb, Hans Wolters, and Gadiel Seroussi, for useful discussions.

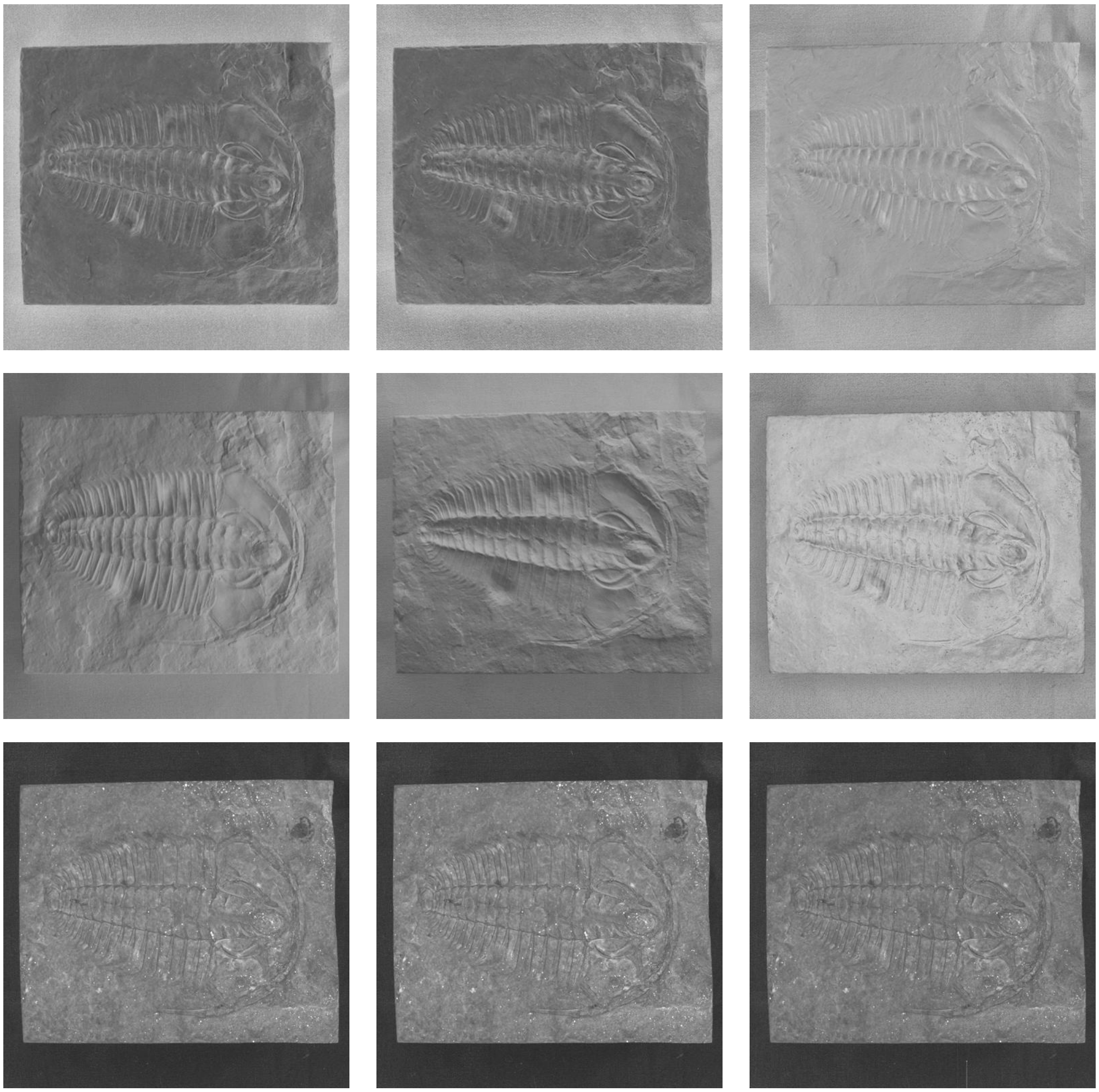


Figure 1: A PTM image used in the experiments (file "trilobite.ptm"). The image format is LRGB. Coefficient planes are displayed from left to right and from top to bottom, in the order a_0^I, a_1^I, a_2^I, a_3^I, a_4^I, a_5^I, followed by the *Red*, *Green* and *Blue* color components.

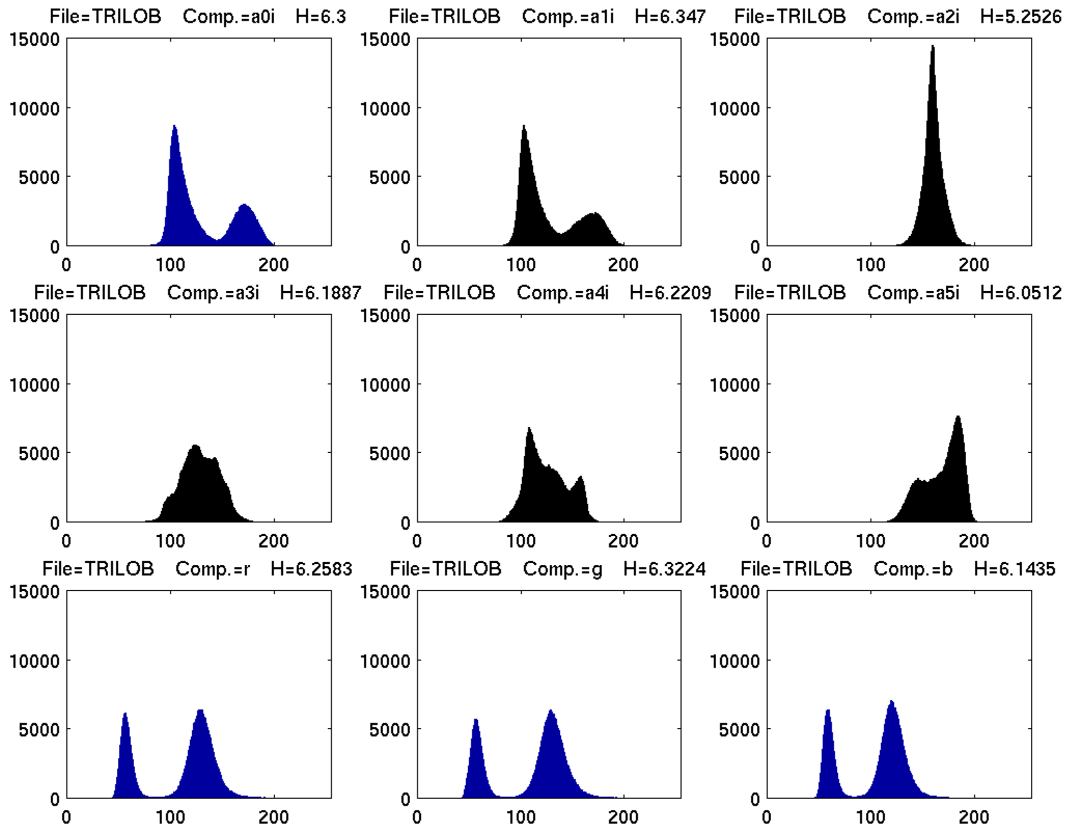


Figure 2: Histograms of coefficient distributions for the PTM file "trilobite.ptm" and the corresponding entropies (in bpt). The image format is LRGB. Coefficient histograms are displayed from left to right and from top to bottom, in the order a_0^I, a_1^I, a_2^I, a_3^I, a_4^I, a_5^I, followed by the *Red*, *Green* and *Blue* color components.

	0	**a0i**	**a1i**	**a2i**	**a3i**	**a4i**	**a5i**	**R**	**G**	**B**
0	Inf	138051	145578	134055	**125445**	129146	148530	151115	153645	149732
a0i	138051	Inf	**131061**	**138048**	138050	138050	138048	138067	138058	138067
a1i	145578	131061	Inf	145575	145577	145576	139267	145604	145594	145604
a2i	134055	138048	145575	Inf	134054	134052	**134050**	134055	134055	134055
a3i	125445	138050	145577	134054	Inf	**125444**	**125444**	125444	**125440**	125444
a4i	129146	138050	145576	134052	125444	Inf	129145	129140	129141	129140
a5i	148530	138048	139267	134050	125444	129145	Inf	144656	148530	144008
R	151115	138067	145604	134055	125444	129140	144656	Inf	**79638**	99309
G	153645	138058	145594	134055	125440	129141	148530	79638	Inf	**98520**
B	149732	138067	145604	134055	125444	129140	144008	99309	98520	Inf

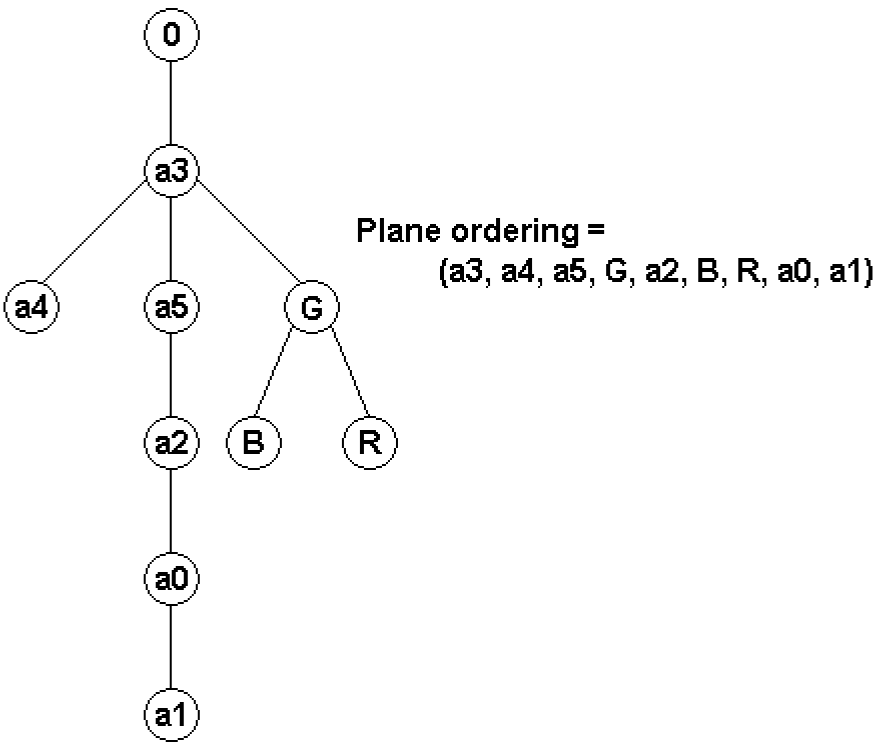
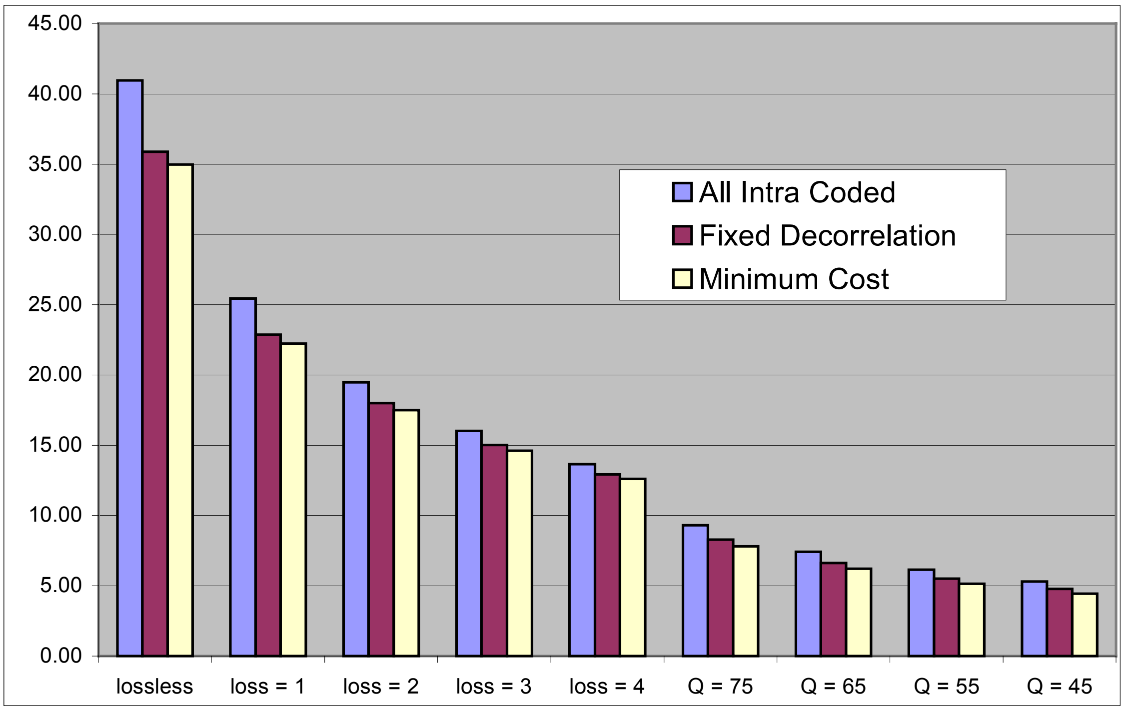


Figure 3: Cost matrix, associated minimum spanning tree, and a compatible plane ordering for the LRGB PTM image “trilobite.ptm”.

	JPEGLS					JPEG				
	lossless	loss = 1	loss = 2	loss = 3	loss = 4	Q = 75	Q = 65	Q = 55	Q = 45	Note
coins	2031567	1241809	986270	851797	740833	386760	283278	215352	173748	RGB
cotton	2106431	1317863	1053185	861544	731712	402618	311089	251826	216530	RGB
mixed	1946743	1204517	975814	843444	753682	434315	350682	295033	257341	RGB
mummy	3586213	2347745	1692872	1238139	966134	596553	452573	363069	316850	RGB
seeds	1441366	949560	814431	730101	662793	372578	303700	254751	222217	RGB
tablet1	1750345	1070855	832043	698090	602077	378500	301536	247465	213186	RGB
tablet2	1715498	1060516	870492	760288	671772	397456	318223	262006	223498	RGB
trilob	1439545	880602	668584	515357	405325	237771	181737	146981	126554	RGB
coins_l	1612965	1153624	950568	836389	747472	506832	412945	345655	298309	LRGB
cotton_l	1842863	1251167	1016295	877086	779644	523039	426572	360554	312153	LRGB
focus2	1231976	735403	557910	433039	341093	248226	204166	174500	157456	1D-RGB
lthouse	479184	282025	207657	167807	140065	130358	112987	101004	93615	1D-RGB
mixed_l	1224239	804790	647222	552794	492833	328311	267567	226931	199072	LRGB
mummy_l	3256099	2059996	1654979	1436623	1281981	780456	629664	524691	451481	LRGB
seeds_l	990605	646305	535851	476514	437054	313924	266165	231259	207793	LRGB
tablet1_l	1107603	698256	548909	469495	413538	265292	213583	176720	150708	LRGB
tablet2_l	1042111	648347	519726	449912	401496	256166	208133	173597	149000	LRGB
trilob_l	1119299	710151	561797	482229	429322	268020	213889	174499	147274	LRGB

	loss = 0	loss = 1	loss = 2	loss = 3	loss = 4	Q = 75	Q = 65	Q = 55	Q = 45	Pixels
coins	62.00	37.90	30.10	25.99	22.61	11.80	8.64	6.57	5.30	262144
cotton	45.71	28.60	22.86	18.70	15.88	8.74	6.75	5.46	4.70	368640
mixed	59.41	36.76	29.78	25.74	23.00	13.25	10.70	9.00	7.85	262144
mummy	34.50	22.59	16.29	11.91	9.30	5.74	4.35	3.49	3.05	831488
seeds	43.99	28.98	24.85	22.28	20.23	11.37	9.27	7.77	6.78	262144
tablet1	53.42	32.68	25.39	21.30	18.37	11.55	9.20	7.55	6.51	262144
tablet2	52.35	32.36	26.57	23.20	20.50	12.13	9.71	8.00	6.82	262144
trilob	43.93	26.87	20.40	15.73	12.37	7.26	5.55	4.49	3.86	262144
coins_l	49.22	35.21	29.01	25.52	22.81	15.47	12.60	10.55	9.10	262144
cotton_l	39.99	27.15	22.06	19.03	16.92	11.35	9.26	7.82	6.77	368640
focus2	20.53	12.26	9.30	7.22	5.68	4.14	3.40	2.91	2.62	480000
lthouse	14.62	8.61	6.34	5.12	4.27	3.98	3.45	3.08	2.86	262144
mixed_l	37.36	24.56	19.75	16.87	15.04	10.02	8.17	6.93	6.08	262144
mummy_l	31.33	19.82	15.92	13.82	12.33	7.51	6.06	5.05	4.34	831488
seeds_l	30.23	19.72	16.35	14.54	13.34	9.58	8.12	7.06	6.34	262144
tablet1_l	33.80	21.31	16.75	14.33	12.62	8.10	6.52	5.39	4.60	262144
tablet2_l	31.80	19.79	15.86	13.73	12.25	7.82	6.35	5.30	4.55	262144
trilob_l	34.16	21.67	17.14	14.72	13.10	8.18	6.53	5.33	4.49	262144
Average	34.98	22.22	17.49	14.60	12.59	7.81	6.22	5.14	4.44	6288128

Table I: Compression results in bits and bits per textel



bits per textel	**lossless**	**loss = 1**	**loss = 2**	**loss = 3**	**loss = 4**	**Q = 75**	**Q = 65**	**Q = 55**	**Q = 45**
All Intra Coded	40.95	25.43	19.47	16.00	13.66	9.30	7.42	6.15	5.30
Fixed Decorrelation	35.88	22.87	18.00	15.01	12.91	8.27	6.62	5.51	4.79
Minimum Cost	34.98	22.22	17.49	14.60	12.59	7.81	6.22	5.14	4.44
% gain									
Fixed Decorrelation	12.39	10.06	7.54	6.25	5.43	11.02	10.78	10.39	9.68
Minimum Cost	2.51	2.84	2.86	2.72	2.48	5.55	6.07	6.70	7.35

Table II: Comparison between PTM test images intra coded, inter and intra coded by using a fixed decorrelation and by finding the decorrelation trough minimization of the cost matrix.

Appendix I: Compressed Polynomial Texture Maps (.ptm) File Format

A compressed PTM file consists of the following 12 sections separated by "newline" characters. When a section consists of multiple elements, represented in ASCII, individual elements are separated by a white space:

1. **Header String**. The ASCII string 'PTM_1.1' appears on the first line of the file. This identifies the file as a PTM file and provides the PTM version number being supported.

2. **Format String**. One of the following ASCII strings appears on the next line identifying the format of the file:

   PTM_FORMAT_JPEG_RGB
   PTM_FORMAT_JPEG_LRGB
   PTM_FORMAT_JPEGLS_RGB
   PTM_FORMAT_JPEGLS_LRGB
   PTM_FORMAT_JPEG2000_RGB
   PTM_FORMAT_JPEG2000_LRGB

   Note that the current version does not support JPEG 2000 compression.

3. **Image Size**. The next line consists of an ASCII string containing the width and height of the PTM map in pixels.

4. **Scale and Bias**. After biasing, scaling, and rounding, each PTM coefficient is stored in the file as a single byte. A total of 6 bias and 6 scale values, one for each of the 6 polynomial coefficients, are provided. The six ASCII floating point scale values appear first in the file, followed by the six ASCII integer biases, all separated by spaces.
   Scale and bias values are taken directly from the original uncompressed PTM file and remain unchanged after compression. As in the uncompressed PTM format, original coefficients are reconstructed according to

$$a_{final} = (a_{raw} - bias) * scale$$

5. **Compression Parameter**. It consists of an ASCII string that contains the parameter being fed to the JPEG-LS or to the JPEG encoder.
   When the file is encoded with JPEG-LS, the parameter represents the lossless mode (if zero) or the maximum absolute value of the loss for each pixel (if greater than zero).
   For JPEG, the parameter represents an encoding quality factor ranging between 20 and 100 (best quality).

6. **Transforms**. It is a sequence of 18 (for RGB PTM's) or 9 (for LRGB PTM's) ASCII integers. The (i+1)-st integer represents the transforms that must be applied to the reference plane (see section 9) before prediction of the coefficient plane indexed by i.

Each transform is represented by a constant value that is a power of 2 (see table below), so in order to specify multiple transforms, constants can simply be OR-ed together to form a single integer. The following transforms are currently implemented:

Transform	**Constant Name**	**Integer Value**
No transform	NOTHING	0
Plane Inversion	PLANE_INVERSION	1
Motion Compensation	MOTION_COMPENSATION	2

7. **Motion Vectors**. It is a sequence of 36 (for RGB PTM's) or 18 (for LRGB PTM's) ASCII signed integers. The first half represents the *x* coordinates and the second half the *y* coordinates of the 18 (or 9) motion vectors.
   Since integers are used to represent half pixel displacements, these values must be divided by 2 in order to obtain the final displacements along the *x* and *y* dimensions.

8. **Order**. It is a sequence of 18 (for RGB PTM's) or 9 (for LRGB PTM's) ASCII integers that represent the order in which the corresponding coefficient plane must be decoded. This order guarantees causality in the decoding process.
   If plane *i* is predicted from plane *j* (called *reference* for *i*), order of *j* will be smaller than order of *i* and decoding of *j* must precede decoding of *i*.
   To start the decoding process, there is always (at least) a plane that is not predicted from any other plane and whose order is 0.

9. **Reference Planes**. A sequence of 18 (for RGB PTM's) or 9 (for LRGB PTM's) ASCII integers that represent the index of the reference plane used for encoding the coefficient plane *i*.
   Coefficient planes are indexed starting from zero: If plane *i* is predicted from plane *j*, then the (*i*+1)-st integer in the sequence is *j*. A special reference index "–1" is used to indicate a plane that is intra coded (i.e., that it is not predicted from any other plane).

10. **Compressed Size**. A sequence of 18 (for RGB PTM's) or 9 (for LRGB PTM's) ASCII integers in which the (*i*+1)-st integer is the size, in bytes, of the *i*-th compressed coefficient plane.
    Since coefficients are not interleaved and compressed planes may have different sizes, this information (and the side information size, see below) must be combined to extract properly the compressed planes from the "Compressed Coefficient Planes" section.

11. **Side Information Size**. A sequence of 18 (for RGB PTM's) or 9 (for LRGB PTM's) ASCII integers in which the (*i*+1)-st integer represents the size (in bytes) of the side information used to correct possible overflows occurring during the (lossy) encoding of the *i*-th coefficient plane (see below). If no overflow occurred during the encoding of the *i*-th plane, the corresponding side information size will be zero.

12. **Compressed Coefficient Planes**. Compressed coefficient planes are stored plane by plane, in sequence, following the original plane ordering. Coefficients are not interleaved and each

plane must be extracted by using the information provided in the sections "Compressed Size" and "Side Information Size." Each compressed plane is stored according to the bit-stream format corresponding to the compression algorithm used (JPEG or JPEG-LS). When a lossy mode is used, each compressed plane is followed by a sequence of zero or more bytes representing the side information necessary to correct overflows resulting from modular arithmetic. This "**Side Information**" section consists of a sequence of pairs (Pixel Position, Pixel Value), represented with five consecutive bytes as follows:

$$(\text{Pixel Position, Pixel Value}) = P_3, P_2, P_1, P_0, V.$$

Pixel position is a 4-byte integer (with the highest order byte stored first), which represents a pixel position when the image is linearized in row order scan, top to bottom, left to right. The pixel value V is the original pixel value that must be substituted in the decoded plane in that position in order to fix the overflow. Overflows must be corrected before using the decoded plane as a prediction reference.

References

[1] T. Malzbender, D. Gelb, H. Wolters and B. Zuckerman, "Enhancement of Shape Perception by Surface Reflectance Transformation", *HP Laboratories Technical Report, HPL-2000-38*, March 2000.

[2] T. Malzbender, D. Gelb and H. Wolters, "Polynomial Texture Map (.ptm) File Format". Private Communication.

[3] M. Weinberger, G. Seroussi and G. Sapiro, "The LOCO-I Lossless Image Compression Algorithm: Principles and Standardization into JPEG-LS", *IEEE Trans. on Image Processing*, Vol. 9, No. 8, August 2000, pp. 1309-1324 (also *HP Laboratories Technical Report, HPL-98-193(R.1)*, November 1999).

[4] W. B. Pennebaker and J. L. Mitchell, *JPEG Still Image Data Compression Standard*, Van Nostrand Reinhold, New York, 1993.

[5] R. Barequet and M. Feder, "SICLIC: A Simple Inter-Color Lossless Image Coder", in *Proceedings of the 1999 Data Compression Conference (DCC'99)*, (Snowbird Utah, USA), pp. 501-510, March 1999.

[6] X. Wu and N. Memon, "Context-Based Lossless Interband Compression – Extending CALIC", IEEE Trans. on Image Processing, Vol. 9, No. 6, June 2000.

[7] B. Carpentieri, M. Weinberger and G. Seroussi, "Lossless Compression of Continuous-Tone Images", *Proceedings of the IEEE*, November 2000 (to appear).

[8] S. Tate, "Band Ordering in Lossless Compression of Multispectral Images", *IEEE Transactions on Computers*, Vol. 46, No. 4, April 1997.