AN ADAPTIVE LIGHTING CORRECTION METHOD FOR MATCHED-TEXTURE CODING

Guoxin Jin[1]*, Thrasyvoulos N. Pappas*[1] *and David L. Neuhoff*[2]

[1]EECS Department, Northwestern Univ., Evanston, IL 60208
[2]EECS Department, Univ. of Michigan, Ann Arbor, MI 48109

ABSTRACT

Matched-Texture Coding is a novel image coder that utilizes the self-similarity of natural images that include textures, in order to achieve structurally lossless compression. The key to a high compression ratio is replacing large image blocks with previously encoded blocks with similar structure. Adjusting the lighting of the replaced block is critical for eliminating illumination artifacts and increasing the number of matches. We propose a new adaptive lighting correction method that is based on the Poisson equation with incomplete boundary conditions. In order to fully exploit the benefits of the adaptive Poisson lighting correction, we also propose modifications of the side-matching (SM) algorithm and structural texture similarity metric. We show that the resulting matched-texture algorithm achieves better coding performance.

Index Terms— Poisson equation, Dirichlet problem, Neumman problem, structural texture similarity metric

1. INTRODUCTION

In order to achieve high compression ratios, image and video compression techniques need to exploit both the signal properties and the characteristics of human perception. One of the keys to further advances in compression is exploiting the redundancy of textured areas, which exhibit a high degree of self-similarity and distortion masking ability [1, 2]. However, traditional techniques fail to do so because, with a few exceptions (e.g., fractal coding [3]), they rely on spatial or temporal prediction coupled with encoding of the residual, which requires relatively high encoding rates, due to the high frequency content of the textured areas. Instead, some recently proposed approaches (as well as an early approach by Popat and Picard [4]) rely on texture analysis/synthesis techniques and transmission of the texture parameters [5, 6, 7, 8, 9]. Another alternative, which we consider in this paper, is the recently proposed matched-texture coding (MTC) approach [10], which exploits texture self-similarity by simply reusing previously encoded texture patches.

One of the key obstacles in exploiting texture self-similarity has been the lack of appropriate texture similarity metrics. Existing approaches rely on PSNR or models of just-noticeable distortion [11] that cannot exploit the stochastic nature of texture, which allows visible point-by-point differences that do not affect the overall quality of the image. However, the development of structural texture similarity metrics [2, 12] opens up new possibilities. MTC [10] attempts to exploit texture self-similarity by simply replacing an image patch (target) with previously encoded patches (candidates) that are structurally similar to – but not perceptually indistinguishable from – the target. If no such match can be found, then the target is encoded with a standard baseline coding technique (e.g., JPEG). To evaluate the similarity between target and candidate patches, MTC uses the new structural texture similarity metric (STSIM-2) [2], which unlike PSNR and perceptual metrics we mentioned above that are point-by-point metrics, relies on local texture statistics. For each successfully replaced target unit (typically 32×32 pixels), only few bits must be transmitted to the decoder. The bits saved in this way can be used for higher quality baseline coding in the other regions to achieve much higher overall rate-distortion performance. The ultimate goal of MTC is structurally-lossless compression [13, 10, 12], whereby the original and reconstructed image are similar but may have visible differences, even though both look natural and are of equally high visual quality.

One of the challenges in achieving low-bitrate structurally-lossless compression with MTC is finding suitable candidates for the target patches and concealing the artifacts that are introduced by patch replacement. There are two kinds of artifacts, spatial discontinuities in texture and illumination. The former can be solved by the use of quilting algorithms [14, 15], while the latter are the main focus of this paper.

In [10], Jin *et al.* introduced a technique that uses interpolation of local average values of the original image to generate a low frequency component (illumination map) that can be subtracted from the original image before MTC is applied. (A similar decomposition into a lowpass lighting and a highpass texture component was used by Ballé *et al.* in their texture coding-by-synthesis approach [9].) However, this approach has three disadvantages. First, the illumination map is computed in a preprocessing step, and is hard to update adaptively during the encoding process. Second, accurately quantizing and encoding of the low frequency illumination map requires a relatively high bitrate. Third, the subtraction of the illumination map introduce significant artifacts in the vicinity of strong edges, which further complicate the encoding process

978-1-4799-2893-4/14/$31.00 ©2014 IEEE

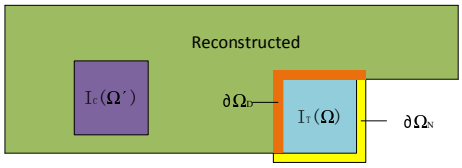

Fig. 1: Boundary Condition

and increase the bitrate.

In this paper, we introduce a new Adaptive Poisson Lighting Correction (APLC) algorithm to eliminate all three disadvantages mentioned above. Moreover, the proposed APLC increases the number of matching candidates for the encoding process. This is because by eliminating the low-pass illuminations, all the blocks with similar structure become suitable candidates. The incorporation of APLC requires a new SM algorithm as well as a modification of the STSIM-2 metric. We show that the proposed additions to MTC result in better quality images and lower bitrate than the algorithm presented in [10]. We also compare with JPEG.

The remainder of the paper is organized as follows. Section 2 reviews Poisson lighting correction. Section 3 introduces the proposed adaptive PLC for MTC compression. A new SM algorithm for MTC and modifications to STSIM-2, along with experimental results are presented in Section 4. The conclusions are summarized in Section 5.

2. POISSON LIGHTING CORRECTION (PLC)

The Poisson equation is a second order partial differential equation (PDE) of elliptic type. For a real/complex function u defined on the domain Ω with boundary $\partial\Omega$, the Poisson equation is defined as follows [16]:

$$-\nabla^2 u = -\nabla^2 f,\ u \in \Omega \tag{1}$$

with one of the following two boundary conditions:

$$u = f,\ u \in \partial\Omega,\ or \tag{2a}$$

$$\frac{\partial u}{\partial \mathbf{n}} = \frac{\partial f}{\partial \mathbf{n}},\ u \in \partial\Omega \tag{2b}$$

where $\nabla^2 u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$ is the Laplacian of u on Ω, $\mathbf{n}$ is the boundary normal, and f is a given scalar function. When the boundary condition in (2a) is used, i.e., the pixel values on $\partial\Omega$ are given, this is known as the Dirichlet problem. When the boundary condition in (2b) is used, i.e., the first-order derivative is given instead of pixel value on $\partial\Omega$, this is called the Neumman problem. However, in some cases (e.g., [17] and the MTC coding problem we discuss below) it is necessary to have mixed boundary conditions. Let the boundary be divided into two distinct, nonoverlapping segments, $\partial\Omega_D$ and $\partial\Omega_N$, such that $\partial\Omega_D \cup \partial\Omega_N = \partial\Omega$, as shown in Figure 1. Then the Dirichlet and Neumann boundary conditions can be applied to $\partial\Omega_D$ and $\partial\Omega_N$, respectively.

A smooth function u satisfying the Poisson equation (1) with boundary condition (2a) or (2b) is called a classical solution [16]. The classical solution is unique if a Dirichlet boundary condition or a mixed boundary condition is given. If only the Neumman boundary condition is available, the solution is not be unique, as adding a constant to u produces another solution. Solving the Poisson equation is equivalent to solving the following minimization problem:

$$\begin{aligned} &\min_u \iint_\Omega |\nabla u - \nabla f|^2,\ s.t. \\ &u = f,\ u \in \partial\Omega_D, \quad \text{and} \quad \frac{\partial u}{\partial \mathbf{n}} = \frac{\partial f}{\partial \mathbf{n}},\ u \in \partial\Omega_N \end{aligned} \tag{3}$$

where $\nabla u = [\frac{u}{x}, \frac{u}{y}]'$ is the gradient of u.

Poisson equations have been used extensively for image impainting, panorama stitching, and texture synthesis [18, 19]. In the highly cited Perez paper [18], only the Dirichlet condition is given, and all the values on $\partial\Omega$ are assumed to be known. In MTC encoding, however, only the left and upper boundaries are available because, assuming the image is encoded in raster scan order, at the time of encoding each target coding unit, only the left and upper boundaries are available. Thus, the use of mixed boundary conditions is necessary.

Sadeghi and *et al.* [17] proposed using mixed boundary conditions when doing image stitching, where $\partial\Omega_D$ is the stitching seam and $\partial\Omega_N$ is the boundary of the image which is going to be lighting corrected. In our experiments, we found that when the lighting of two images is significantly different, it is hard to find a good solution with such mixed boundary conditions. To avoid this problem, in the next section we propose a modified PLC algorithm, in which, in addition to the available Dirichlet boundary condition on $\partial\Omega_D$ and the Neumman boundary condition on $\partial\Omega_N$ (zero derivative in the direction normal to the boundary), we introduce additional Dirichlet boundary conditions on $\partial\Omega_N$. Once the lighting correction is done, we use thin-plate spline smoothing to get an explicit estimate of the lighting, which we can then compare to the lighting of the original image.

Thin-Plate Spline Smoothing (TPSS): A thin-plate spline is a second order spline [20, 21] that is widely used for 2D signal regression and smoothing, and is defined as

$$\min_v \lambda \textstyle\sum_\Omega ||u - v||^2 + (1-\lambda) \iint_\Omega \left[\frac{\partial^2 v}{\partial x^2} + 2\frac{\partial^2 v}{\partial x \partial y} + \frac{\partial^2 v}{\partial y^2}\right] dxdy \tag{4}$$

where $u \in \Omega$ is the input image, v is the smoothed function, and λ controls the smoothness. If $\lambda \to 1$, (4) becomes least squares regression, and if $\lambda \to 0$, a second order smoother.

3. ADAPTIVE PLC (APLC)

As we discussed in Section 2, when the lighting of the target and surrounding blocks are significantly different, PLC with mixed boundary conditions (3) could fail. Thus, we introduce an additional Dirichlet boundary condition on $\partial\Omega_N$. The motivation comes from the predictive quadtree interpolation (PQI) coder [22], which encodes the "foot" (lower right pixel) of a target block and uses it, along with the known pixels in the left and upper boundary of the block, to interpolate the other values in the block. In PQI, a special metric is

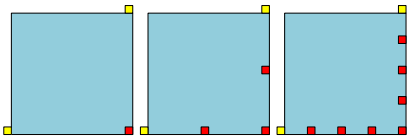

Fig. 2: Feet locations

used to measure the similarity between the interpolated target block and the original block. If the interpolated block is not sufficiently similar to the original, the target block is split into four subblocks, and the procedure is repeated for each subblock until an adequate match is found or a minimum block size is reached. Thus, a quadtree is formed.

For the boundary conditions on $\partial\Omega_N$, we adopt a similar approach. However, in PQI, the interpolation is intended to succeed when the block is relatively smooth. In contrast, our goal is to estimate the lighting of textured blocks. Thus, the value of a single pixel is not suitable for estimating the foot. Instead, we encode the local average of the image at the foot for a more accurate estimate. For similar reasons, we use the local average of the known pixels near the upper right and lower left corners of the target block, which along with the foot (or feet), are used to linearly interpolate the remaining values of the boundary. Moreover, additional feet along the lower and right boundary can be used to provide more accurate estimates of the lighting. Figure 2 shows the possible locations of the encoded feet as red pixels and the known corner average values as yellow pixels.

We now define a modified PLC (MPLC) approach, which will become adaptive (APLC), when we allow a variable number of feet. Let $I_T(\Omega)$ be the original image block to be coded (called target) and $I_C(\Omega')$ be the candidate block which will replace I_T in the coded image. We use $\hat{I}_T$ to denote the interpolated boundary on $\partial\Omega_N$that is based on the encoded feet. Then the Poisson equation becomes:

$$-\nabla^2 u(\Omega) = -\nabla^2 I_C(\Omega'),\ u \in \Omega,\ \text{with boundary conditions:}$$
$$u = I_T, u \in \partial\Omega_D; \frac{\partial u}{\partial \mathbf{n}} = 0, u \in \partial\Omega_N; u = \hat{I}_T, u \in \partial\Omega_N \quad (5)$$

PQI encodes only one foot at the lower right corner pixel of the block. Since the rest of the block is interpolated, providing additional feet is equivalent to subdividing the block, which is what PQI does. In MTC, however, the feet are used to interpolate the boundary conditions, while the remaining pixels are obtained based on the candidate block and the solution of the Poisson equation with mixed boundary conditions. Since the coding gains increase with increasing block size, it pays to use (an adaptive number of) additional feet in order to obtain more accurate lighting estimates. A block can have zero (no lighting correction), one, or multiple feet. The error criterion for selecting the number of feet is illustrated in Figure 3. We compute the mean squared error (MSE) between the TPSS of the original target block and the TPSS of the MPLC based on the target block and interpolated feet as specified in (5).

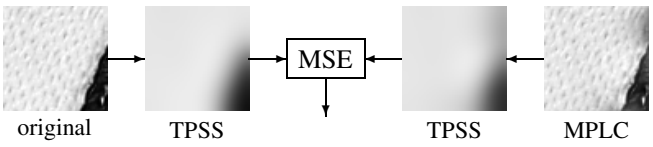

Fig. 3: Error Measurement for selecting number of feet

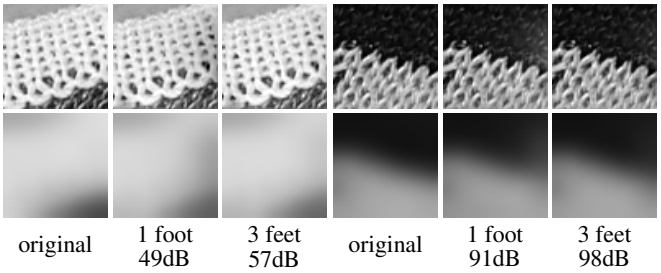

Fig. 4: Modified PLC Examples

To demonstrate the need for additional feet, we consider the ideal case where an image block is used as both the target and the candidate. Figure 4 shows two examples of an original block and the MPLC with one and three feet (top row) and the corresponding TPSS (bottom row). Figure 5 demonstrates how MPLC works on the entire image for a fixed block size (64×64 pixels) and a fixed number or feet (1 and 3). Note, however, that the original image block is used as both the target and the candidate. TPSS is then applied to the original image and the MPLC of each block, and the PSNR is computed. Figures 4 and 5 demonstrate that the additional feet result in considerable improvement in lighting correction performance. Note, however, that one foot is sufficient for the homogeneous regions of the image, while more feet are needed in more complex transition regions. Thus, an adaptive number of feet is needed.

The error criterion we discussed above can be used to select the minimum number of feet that meet a given PSNR requirement. Figure 6 shows the APLC results for variable block size. If the requirement is not met with the maximum number of feet (typically three), then the block is subdivided into four subblocks. Note that in Figure 6, if a minimum block size is reached and the criterion is not met, the failed APLC results are still shown as noticeable artifacts. In the actual coder implementation, however, these blocks will be encoded with the baseline encoder. We should also point out that in this example, since we use the original image block as both target and candidate, using no lighting correction is always the best. Thus, in order to test the performance of APLC, we had to force at least one foot. Figure 6 shows examples with two initial block sizes (64×64 and 128×128), and also compares with the lighting correction approach of [10].

Figure 7 plots the rate-distortion curves of APLC with different number of feet and different block sizes. It also shows APLC with variable block sizes and number of feet for different initial block sizes. The curves are obtained by varying the PSNR threshold for the lighting correction. Note the superior

TPSS on original | 1 foot, 28.9 dB 0.001bpp | 3 feet, 32.2 dB 0.0035bpp

Fig. 5: MPLC with fixed block size (64×64)

initial size 64×64 32.2 dB, 0.007 bpp | initial size 128×128 37.0 dB, 0.0057 bpp | lighting correction in [10]

Fig. 6: APLC with variable block size vs. LC in [10]

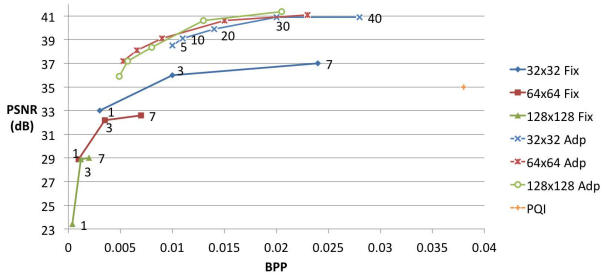

Fig. 7: Rate-Distortion analysis

rate-distortion performance of the adaptive approach. Looking at Figures 5 and 6, one can conclude that the adaptation results in better visual quality, too. The initial block size does not seem to have a strong effect.

Post PLC for extra boundary conditions: One of benefits with adaptive lighting correction algorithm available for the coder is that it can use all of the available information. Thus, if an image block is encoded with MTC and the adjacent blocks to the right or below are encoded with the baseline coder, then the lighting of the block can be updated to incorporate the additional boundary information. This does not cost any bits, only computation time.

4. MODIFIED MTC (MTC-2)

As we discussed, the incorporation of APLC requires a new SM algorithm. Given the L-shaped boundary B_t of the target block, we find the candidate boundary B_c in the previously encoded region that minimizes the function:

$$\min_{B_c} \left\{ \min\left[||B_t - B_c||_2/t_1, 1\right] + \lambda \min[log \left| \frac{v_c}{v_t} \right| /t_2, 1] \right\} \quad (6)$$

where v_t and v_b are the variances of B_t and B_c, respectively, and t_1 and t_2 are thresholds. The first term in (6) represents a pixel-based quality measurement and the second a structural quality measurement. The normalization and clipping are used to compare the two terms in the same dimension.

JPEG | Original MTC | MTC with APLC

Fig. 8: Comparison with previous results at 0.34 bpp

The parameter λ controls the effectiveness of the structural constraint. Typical values are in $[0.1, 1]$. When λ is large, SM emphasizes texture similarity and deemphasizes the macro structure such as strong edges. We found that $\lambda = 0.5$ is a reasonable option. Moreover, in order to find more candidates, the first term in (6) can be computed after the mean is subtracted.

To fully utilize the advantages of adaptive lighting correction, we must also modify the luminance term L of STSIM-2 [2], which is sensitive only to big illumination differences. Instead, we use the following:

$$L = \min\left\{|\mu_x - \mu_y|^2/t_3, 1\right\} \quad (7)$$

where μ_x and μ_y are the means of the target and candidate coding units, respectively, and t_3 is a clipping threshold. The smaller t_3 is, the more sensitive L is to illumination change. Typical values of t_3 are 1 to 4.

Our experimental results show a significant reduction in lighting discontinuity artifacts when APLC is incorporated in MTC compared to the previous approach [10], which used a combination of SM and direct block matching (DBM). More importantly, the proposed scheme comes at considerably lower computational complexity (10 minutes versus 4 hours). Figure 8 shows the new coding result using APLC with constrained SM and modified STSIM-2, and compares it with the result of [10] and JPEG at the same bit rate. It can be seen that the proposed algorithm works much better in the regions where texture and texture edges are present, demonstrating the advantages of the new lighting correction approach. In addition, in texture regions, e.g., the sweater and hair, the proposed method reuses more texture blocks than [10]. On the other hand, in smooth regions, the proposed approach may introduce artifacts near weak edges, as can be seen around the chin and thumb. This is due to incorrect matches, not the lighting correction. Compared to JPEG, the proposed approach provides better overall visual reconstruction quality at the same bitrate.

5. CONCLUSIONS

We have proposed a new adaptive Poisson lighting correction algorithm for MTC. It is based on the solution of a Poisson equation with a new boundary condition that consists of coded and interpolated image values that save bits while maintaining perceptual quality. We also introduced a new side matching algorithm and a modification to the lighting term of STSIM-2. The experimental results show a significant increase in the perceptual quality of the resulting MTC.

6. REFERENCES

[1] A. C. Brooks, X. Zhao, and T. N. Pappas, "Structural similarity quality metrics in a coding context: Exploring the space of realistic distortions," *IEEE Trans. Image Process.*, vol. 17, pp. 1261–1273, Aug. 2008.

[2] J. Zujovic, T. N. Pappas, and D. L. Neuhoff, "Structural texture similarity metrics for image analysis and retrieval," *IEEE Trans. Image Process.*, vol. 22, pp. 2545–2558, July 2013.

[3] A. E. Jacquin, "Image coding based on a fractal theory of iterated contractive image transformations," *IEEE Trans. Image Process.*, vol. 1, pp. 18–30, Jan. 1992.

[4] K. Popat and R. W. Picard, "Novel cluster-based probability model for texture synthesis, classification, and compression," in *Proc. SPIE Visual Communications*, (Cambridge, MA), 1993.

[5] P. Ndjiki-Nya, D. Bull, and T. Wiegand, "Perception-oriented video coding based on texture analysis and synthesis," in *Proc. Int. Conf. Image Processing (ICIP))*, pp. 2273–2276, Nov. 2009.

[6] S. Ierodiaconou, J. Byrne, D. R. Bull, D. Redmill, and P. Hill, "Unsupervised image compression using graphcut texture synthesis," in *Proc. Int. Conf. Image Processing (ICIP))*, pp. 2289–2292, Nov. 2009.

[7] M. Bosch, F. Zhu, and E. J. Delp, "Segmentation-based video compression using texture and motion models," *IEEE J. Sel. Topics Signal Process.*, vol. 5, pp. 1366–1377, Nov. 2011.

[8] F. Zhang and D. R. Bull, "A parametric framework for video compression using region-based texture models," *IEEE J. Sel. Topics Signal Process.*, vol. 5, pp. 1378–1392, Nov. 2011.

[9] J. Balle, A. Stojanovic, and J.-R. Ohm, "Models for static and dynamic texture synthesis in image and video compression," *IEEE J. Sel. Topics Signal Process.*, vol. 5, pp. 1353–1365, Nov. 2011.

[10] G. Jin, Y. Zhai, T. N. Pappas, and D. L. Neuhoff, "Matched-texture coding for structurally lossless compression," in *Proc. Int. Conf. Image Processing (ICIP)*, (Orlando, FL), pp. 1065–1068, Oct. 2012.

[11] T. N. Pappas, R. J. Safranek, and J. Chen, "Perceptual criteria for image quality evaluation," in *Handbook of Image and Video Processing* (A. C. Bovik, ed.), pp. 939–959, Academic Press, second ed., 2005.

[12] T. N. Pappas, D. L. Neuhoff, H. de Ridder, and J. Zujovic, "Image analysis: Focus on texture similarity," *Proc. IEEE*, vol. 101, pp. 2044–2057, Sept. 2013.

[13] T. N. Pappas, J. Zujovic, and D. L. Neuhoff, "Image analysis and compression: Renewed focus on texture," in *Visual Information Processing and Communication*, vol. 7543 of *Proc. SPIE*, (San Jose, CA), Jan. 2010.

[14] A. A. Efros and W. T. Freeman, "Image quilting for texture synthesis and transfer," in *Proc. 28th Intl. Conf. Computer Graphics and Interactive Techniques (SIGGRAPH-01)*, (Los Angeles, CA), pp. 341–346, Aug. 2001.

[15] V. Kwatra, A. Schödl, I. Essa, G. Turk, and A. Bobick, "Graphcut textures: Image and video synthesis using graph cuts," *ACM Transactions on Graphics, SIGGRAPH*, vol. 22, no. 3, pp. 277–286, 2003.

[16] H. C. Elman, D. J. Silvester, and A. J. Wathen, *Finite Elements and Fast Iterative Solvers with Applications in Incompressible Fluid Dynamics*. Oxford University Press, first ed., 2005.

[17] M. A. Sadeghi, S. M. M. Hejrati, and N. Gheissari, "Poisson local color correction for image stitching," in *VISAPP (1)*, pp. 275–282, 2008.

[18] P. Pérez, M. Gangnet, and A. Blake, "Poisson image editing," in *ACM SIGGRAPH 2003 Papers*, SIGGRAPH '03, (New York, NY, USA), pp. 313–318, ACM, 2003.

[19] A. Levin, A. Zomet, S. Peleg, and Y. Weiss, "Seamless image stitching in the gradient domain," in *Proc. European Conf. Computer Vision (ECCV)*, 2006.

[20] R. Franke, "Smooth interpolation of scattered data by local thin plate splines," *Computers & Mathematics with Applications*, vol. 8, no. 4, pp. 273–281, 1982.

[21] M. J. Buckley, "Fast computation of a discretized thin-plate smoothing spline for image data," *Biometrika*, vol. 81, no. 2, pp. 247–258, 1994.

[22] C.-Y. Teng and D. L. Neuhoff, "Quadtree-guided wavelet image coding," in *Proc. IEEE Data Compression Conf.*, (Snowbird, Utah), pp. 406–413, Apr. 1996.