Building Structural Similarity Databases for Metric Learning

Guoxin Jin and Thrasyvoulos N. Pappas,
Department of Electrical Engineering and Computer Science,
Northwestern University,
Evanston, IL, 60201

March 19, 2015

ABSTRACT

We propose a new approach for constructing databases for training and testing similarity metrics for structurally lossless image compression. Our focus is on structural texture similarity (STSIM) metrics and the matched-texture compression (MTC) approach. We first discuss the metric requirements for structurally lossless compression, which differ from those of other applications such as image retrieval, classification, and understanding. We identify "interchangeability" as the key requirement for metric performance, and partition the domain of "identical" textures into three regions, of "highest," "high," and "good" similarity. We design two subjective tests for data collection, the first relies on ViSiProG to build a database of "identical" clusters, and the second builds a database of image pairs with the "highest," "high," "good," and "bad" similarity labels. The data for the subjective tests is generated during the MTC encoding process, and consist of pairs of candidate and target image blocks. The context of the surrounding image is critical for training the metrics to detect lighting discontinuities, spatial misalignments, and other border artifacts that have a noticeable effect on perceptual quality. The identical texture clusters are then used for training and testing two STSIM metrics. The labelled image pair database will be used in future research.

1. INTRODUCTION

Texture similarity metrics are important for a variety of applications, including image compression, restoration, content-based retrieval, and semantic information extraction.[1] Deriving texture similarity metrics that agree with human perception is quite challenging because of the stochastic nature of most textures and the ability of the human visual system (HVS) to perceive textures with significant point-by-point distortions as very similar or virtually identical. As a result, traditional image similarity metrics (commonly referred to as image quality metrics) cannot effectively quantify texture similarity. A recently proposed class of structural texture similarity metrics (STSIMs)[2] has shown a lot of promise, but a lot more work is needed to optimize their performance in the context of demanding applications, such as image compression.

The focus of this paper is on *structurally lossless image compression,* whereby in a side-by-side comparison an original and a compressed image may look different but have similar quality, essentially the same content, and one cannot tell which is the original.[1,3,4] An example is shown in Figure 1; both of the baboons look natural, but 20% of the pixels are different and the PSNR is 22.2 dB. An approach for achieving structurally lossless compression, called matched texture coding (MTC), was proposed by Jin *et al.* in Refs. 1, 4. However, even though it has been shown that for a certain range of bitrates MTC can outperform traditional compression techniques such as JPEG,[4–6] a much higher compression ratio is desirable. The texture similarity metric has been identified as the bottleneck for obtaining higher compression efficiencies.

The goal of this paper is to construct databases for testing and training STSIMs in the context of structurally lossless compression. In particular, we will identify the desired performance requirements, and based on those, we will construct a database that can be used to learn the parameters of specific metrics. The existing STSIMs,[2,7,8] as well as the structural similarity metrics (SSIMs)[9–11] that inspired their introduction, compute and combine a variety of features (some of which have been perceptually motivated) in virtually heuristic ways. A more systematic approach for metric training was proposed by Maggioni *et al.* in Ref. 12, whereby the STSIM metric parameters are optimized in the content-based retrieval context with the intention of also using the metric in image restoration and compression. We will adopt and further optimize the approach of Ref. 12, but we will

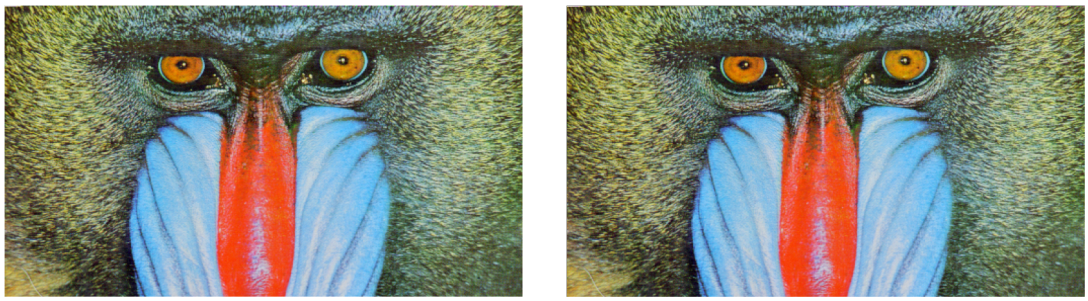

Figure 1. Example of structural similarity

also assume a more general form of STSIM, with the goal of optimizing its parameters based on the proposed database construction.

In order to perform supervised or semi-supervised learning for texture similarity metrics, there is a need to build a database of texture pairs, each of which is associated with a quantitative subjective measurement (score) that indicates the similarity of the pair. While our primary focus is on textures, the metric will be embedded in a structurally lossless application that can be applied to arbitrary image content. Thus, in addition to pure textures, the database must include patches that are mixed, that is, contain edges between textures and between textures and smooth regions, as well as smooth regions and edges between smooth regions. Furthermore, the databases must include patches with coding distortions.

A variety of databases for metric training can be found in the literature. The LIVE,[13] TID2008, and TID2013[14] databases are aimed at general image compression applications and are widely used. However, they consider only distortions of the entire image, and as such cannot be used for STSIM metric training. Several databases are used by state-of-the-art metric learning approaches are designed for other applications, like face and handwriting recognition,[15–17] face recognition,[18] hand-writing recognition,[19] voice recognition,[20] etc.

Among databases focused on texture, the CUReT (Columbia-Utrecht Reflectance and Texture) database[21] contains 61 surfaces with different BRDF and BTF. Among other applications, this has been used by Zujovic *et al.*[2] for testing STSIMs in the context of content-based retrieval; Zujovic *et al.*[2] also used a database containing Corbis images. The Outex texture database[22] contains 28 classes of 320 surface images with different illumination, resolution, and rotation angles. This, too, is aimed at content-based retrieval. The MIT VisTex database[23] contains natural texture patches as well as real-world scenes containing textures with different perspectives and illumination conditions, and is aimed at computer vision applications. The BTF (bidirectional texture function) database of Ref. 24 contains synthesized textures of materials under different lighting conditions, and is aimed at material classification. The Kylberg texture dataset[25] contains texture images of fabrics, stones, grains, etc. and is also aimed at texture classification and recognition. Finally, the Prague texture segmentation data generator[26] is aimed at texture segmentation.

All of the texture databases we discussed so far are aimed at retrieval and recognition, and thus, the ground truth is in binary form (a pair of textures belongs to the same class or not). The only attempt to design a database for texture similarity in coding applications was by Zujovic *et al.*,[27] who generated synthetic distortions of natural texture images using micro (local) translations, rotations, and warping. The advantage of synthetic distortions is that one can easily control the degree of distortion for testing and training. The goal of our work is to complement this type of data with real textures that arise in the context of compression applications. Our subjective test presents the subjects with patches embedded in the context (about 5 times patch size in each direction). This is important because in the structurally lossless compression, the interchangeable patches must not only be similar but also aligned well to the context so that few visible (spatial discontinuity, illumination/lighting change, and JPEG coding) distortions exist.

In the remainder of this paper, in Section 2 we briefly review STSIMs and MTC. In Section 3 we discuss the requirements for metric performance, while in Section 5 we provide a detailed description of the subjective experiments. Section 6 discusses metric training and Section 7 summarizes the conclusions.

2. BACKGROUND

In this section we briefly review STSIM metrics, focusing on the ones most relevant to this paper, and the MTC coder.

2.1 Structural Texture Similarity Metrics

The main idea in the development of STSIM metrics is to replace traditional point-by-point comparisons with comparisons of local statistics computed in sliding windows at corresponding locations in the two images.[2] The original inspiration came from the SSIMs,[28] which actually still include the structure term, which is point-by-point. Our focus will be on the STSIM-M metric[2] and its variations that are the most relevant for this work. All STSIM metrics are computed in the subband domain. We will assume a steerable filter decomposition[29] with 3 scales and 4 orientations.

Let $\boldsymbol{x}$ and $\boldsymbol{y}$ be two patches at corresponding window locations and corresponding subbands of two images. The STSIM metrics compute a number of statistics in these patches that include, the mean, variance, horizontal and vertical correlation coefficients, as well as cross-correlation coefficients across radially adjacent subbands with the same orientation and across all orientations of the same scale. The total number of statistics is 82 (56 if we drop the cross-band correlations). The STSIM metrics then compare these statistics and pool the results over subbands and window locations to obtain a value that represents the overall image similarity. While, the statistic comparisons and subband pooling of STSIM-1 and STSIM-2 are more or less fixed and rather heuristic, the STSIM-M (Mahalanobis) is the most intuitive and offers the most flexibility. It constructs vectors that contain all the statistics for all the subbands of each image patch, and then computes the Mahalanobis distance between the vectors. Let $\boldsymbol{\omega}_x = \{\omega_{x,k}\}$ and $\boldsymbol{\omega}_y = \{\omega_{y,k}\}$ be the vector of statistics (feature vectors) the patches $\boldsymbol{x}$ and $\boldsymbol{y}$, then if we assume that the subbands are not correlated, the STSIM-M is:

$$Q_{\text{STSIM-M}}\boldsymbol{x}, \boldsymbol{y} = \sqrt{\sum_{k=1}^{N_p} \frac{(\omega_{x,k} - \omega_{y,k})^2}{\sigma_k^2}}, \tag{1}$$

where k is the index of the subband statistic and σ_k^2 is its variance over all of the training data. However, in the most general case, where the subbands may be correlated, the metric becomes:

$$Q_{\text{STSIM-M2}}(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{\sum_{l=1}^{N_p}\sum_{l=1}^{N_p} \frac{(\omega_{x,k} - \omega_{y,k})(\omega_{x,l} - \omega_{y,l})}{\sigma_{k,l}}}, \tag{2}$$

where $\sigma_{k,l}^2$ is the covariance of the k-th and l-th subband statistics over the training data.

STSIM-M offers the ability of giving different weights to different statistics; the statistics with large variance are de-emphasized. This makes sense if the variance is due to noise, but does not when the variance is due to intra-class variations of each statistic. To sort out these cases, Maggioni *et al.*[12] proposed STSIM-I (intra), a variation of STSIM-M[2] that computes the variance of statistic k within each class, and then averages the variances over all classes for all the patches to obtain ς_k^2, which represents the variance due to intra-class variations. A large intra-class variance makes the statistic less informative, as opposed to large inter-class variations, which make the statistic more informative:

$$Q_{\text{STSIM-I}} = \sqrt{\sum_{k=1}^{Np} \frac{(\omega_{x,k} - \omega_{y,k})^2}{\varsigma_k^2}}. \tag{3}$$

Note that there is no need to explicitly incorporate the inter-class variance in the metric, as large inter-class variations do result in higher metric values.

The goal will be to construct databases for the proper training of both STSIM-I and STSIM-M2.

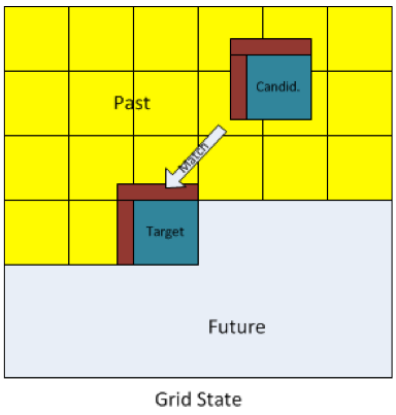

Figure 2. Matched-texture coding

2.2 Matched-Texture Coding (MTC) for Structural Lossless Compression

In MTC, an image is partitioned into rectangular (typically square) blocks, which are coded in raster scan order, as shown in Figure 2. Each block is encoded either by a baseline coder like JPEG or by replacing it with a previously encoded block. Thus, MTC is different from state-of-the-art video coding techniques, which rely on motion compensation[30] or intra block "copy" prediction,[31] and typically encode a residual (the difference between the original and the prediction or "copy"). There are no residuals to be encoded in MTC. We will refer to the block to be encoded as the target block, and a block that we consider replacing it with as the candidate block. The goal is to find candidate blocks that are structurally similar to the target, so that when they replace the target, the quality of the image remains the same. To avoid encoding the location of the candidate block, MTC selects a number of candidate blocks whose L-shaped context (the brown region in Figure 2) matches the context of the target block. Then, it uses an STSIM metric to find which of these candidates best matches the target block. If there are K candidates, then this requires only $\log_2 K$ bits. If no candidate matches the target, then the block is either subdivided into four blocks, or if a minimum block size is reached (typically 16×16), the block is encoded by the baseline coder. (This requires an additional bit.) For each successfully replaced target block (typically 32×32 but the bigger the size the more the savings), only a few bits are required to transmit to the decoder. The bits saved can be used for higher quality baseline coding of the rest of the image, in order to achieve higher rate-distortion performance. Finally, to avoid blocking artifacts, texture blending is applied in the block border areas. These are the key MTC ideas. More details can be found in Refs. 4–6.

The key to achieving structurally-lossless compression with MTC is that the STSIM metric can help identify blocks that are structurally equivalent to the target blocks. However, the existing STSIMs, which have achieved impressive results for image retrieval,[2] are not as effective when used in conjunction with MTC. The problem is that for some of the matches the target and the candidate are not *interchangeable*. This is because the metric sometimes will accept matches that are (visually) not that good or it will favor a bad match over a better one.

3. DESIRED PERFORMANCE FOR STSIM METRICS

Before we design and construct a database for metric training, we must understand the requirements for metric performance. As the authors of Refs. 1, 32 point out, image compression requires different metric performance than other applications such as image retrieval and image classification. While in retrieval and classification it may be sufficient to distinguish between similar and dissimilar textures, in image compression it is important to ensure a monotonic relationship between measured and perceived distortion.

In Ref. 1, 32, a conceptual plot of the desired relationship between subjective texture similarity and metric values is proposed, and is reproduced here in Figure 3. For a metric to be able to distinguish between similar and dissimilar pairs of textures, all that is needed is for the metric to give low values for dissimilar pairs and high values for similar pairs. A monotonic relationship between objective and subjective scores is only needed (and

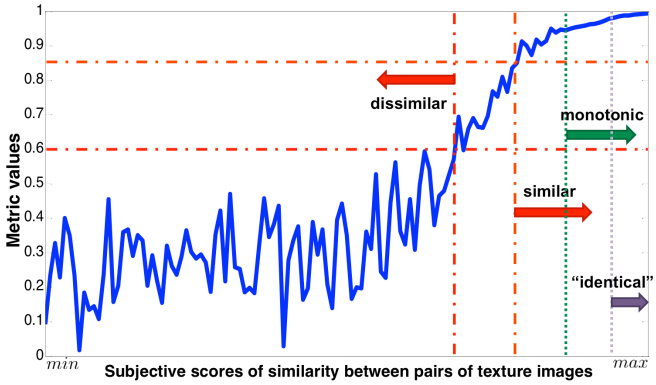

Figure 3. Schematic illustration of desired STSIM performance[32]

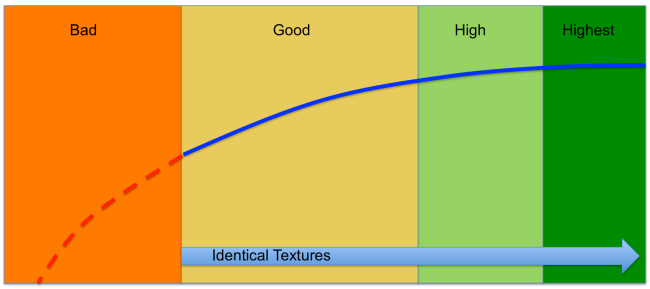

Figure 4. Desired STSIMs performance beyond identical texture domain

only attainable) when the similarity is very high, that is, the textures are almost the same or one is a mildly distorted version of the other.

Zujovic and co-authors[1,2,32] also identify the need for detecting *identical* textures, which means that they could be pieces of the same large perceptually uniform texture; this is also important for image retrieval and classification. However, our research shows that even if the target and the candidate blocks are identical textures, replacing one with the other may not result in structurally lossless compression, as any misalignment, lighting discontinuity, or other border effect will create a noticeable artifact that reduces the perceptual quality. In principle, such artifacts can be avoided with good blending and lighting correction algorithms,[4–6] but these algorithms are not always perfect, plus the metric should be able to identify the artifacts anyway. We will refer to this stricter metric requirement for structurally lossless compression as the ability to detect *interchangeable* textures.

The focus of this paper is thus in the domain of interchangeable textures, which is beyond (to the right of) the domain identical textures in Figure 3. For the purposes of structurally lossless compression, we further partition the identical texture domain to three regions, as shown in Figure 4. In the *highest* similarity range, the textures can be interchanged without any loss of perceptual quality (structurally lossless). In the *high* similarity range, the textures can be interchanged but will introduce a small loss of perceptual quality (not quite structurally lossless but acceptable). In the *good* similarity range, the textures are identical textures, but interchanges will result in significant loss of perceptual quality (structurally lossy and unacceptable), that manifests itself in misalignments, lighting discontinuities, and other noticeable artifacts. For the purposes of structurally lossless compression, any textures that are not in the identical texture domain (good, high, and highest range), will belong to the *bad* similarity range.

One very important difference between interchangeable textures and identical textures is that the transitive property is not maintained. In other words, if texture A is interchangeable with texture B and texture B is interchangeable with texture C in their coding context, it is not guaranteed that A is interchangeable with C. Thus, one cannot do clustering in this situation.

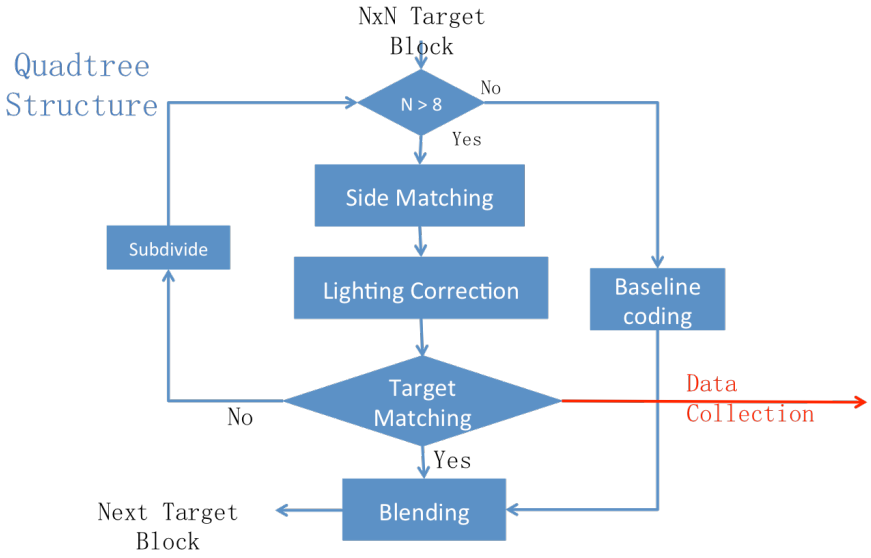

Figure 5. MTC Flowchart

4. BUILDING STRUCTURAL SIMILARITY DATABASE

As we discussed in previous sections, the goal of our work is to complement existing metric training data with real textures that arise in the context of compression applications, and specifically, MTC. Thus, we ask the subjects to evaluate the same texture pairs that the MTC coder generates during the encoding process. Of course, then MTC uses an STSIM to find the candidate that best matches a given target and whether it is acceptable as a target replacement. The flowchart of the MTC coder is shown in Figure 5. The side matching (SM) identifies the candidate blocks based on either MSE[5] as the similarity metric or a combination of statistical similarity metrics and MSE;[5,6] however, STSIM is used to make the critical decision whether the candidate matches the target, after the the lighting correction is performed. So, this is the point where the data collection for the test must take place. We have identified two types of subjective tests.

Subjective Test I: The goal of this test is to form clusters of identical textures. For the formation of the clusters we rely on Visual Similarity Progressive Grouping (ViSiProG).[32] The metric should then be able to separate the clusters, that is, it should give high values for textures that belong to the same cluster and low values for textures that belong to different clusters. The advantage of this test is that we can collect a lot of data in relatively easy fashion for extensive metric testing. Note that since the data is binary (a pair of textures is identical or not), this test does not solve the interchangeability problem. It does, however, help reject the majority of candidates that are not suitable for replacing the target. Figure 6 (left) shows the structure of the texture space after the cluster formation. Note that the nature of cluster formation guarantees that the clusters will be distinct.

Subjective Test II: The goal of this test is to collect data for fine tuning the selection of the best candidate block for replacing the target, so that the interchangeability requirement will be satisfied. The subjects are given a pair of candidate and target textures and are asked to give a rating of highest, high, good, and bad for the similarity of the pair. It is important that the comparison is made in the context of the image being compressed, that is, that each block is surrounded by the corresponding image context. This is in order to take into account the effects of misalignments, lighting discontinuities, and other noticeable artifacts. The details will be discussed in the next section. The data will be used to assign numeric subjective similarity values to the texture pairs in the databases, which will then be used for metric training. Figure 6 (right) provides a graphical representation of the metric space after the data collected during this test. For any target texture block, we obtain three concentric ellipsoids centred at the target block. The types of similarity are represented by the concentric ellipses. Note

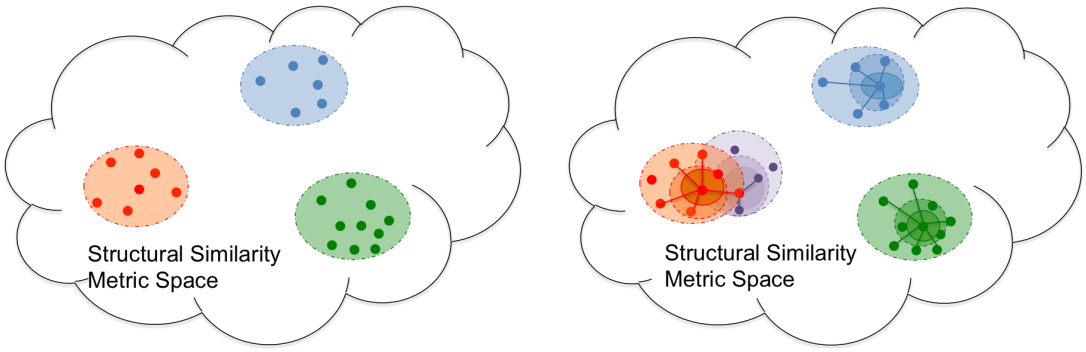

Figure 6. Structural similarity metric space after Subjective Test I (left) and after Subjective Test II (right)

that each target block will have each own ellipse. Overlaps are possible but there is no merging of the ellipses, as indicated in the figure. This is consistent with the lack of a transitive property we discussed above.

Finally, we should point out that both subjective tests are carried out with grayscale images, as we want to separate the study of structural similarity from color similarity.

5. DETAILED DESCRIPTION OF THE SUBJECTIVE TESTS

We now discuss the detailed experimental setup. As we saw in the previous section, the will use MTC as the primary tool for generating the images for the database and the associated subjective tests. At the target matching step of MTC, there are K candidate blocks to be compared with the target block. These candidates are selected by side matching. In this paper we use the MSE constrained by the log variance ratio (LVR)[5] as the side matching similarity metric and the Poisson lighting correction.[5] Instead of using STSIM, the target matching is done by the subject, so that the coding can proceed. However, the comparisons of the K candidates ($K = 8$ in this paper) also form the basis for out two tests. Throughout this paper, we assume a fixed block size. If a coder uses a range of block sizes, then a separate database will be formed for each size.

5.1 Subjective Test I: ViSiProG-based Clustering

For this test, at the target matching stage of MTC, we collect all the target and candidate blocks in the database. We used four 1024×1024 images for this experiment. The image blocks are then clustered using ViSiProG.[32] ViSiProG is an efficient tool for conducting subjective experiments to organize an image database into clusters of visually similar images. Each subject forms groups of a fixed number (typically 9) of similar images in a progressive fashion, seeing a subset of images (say 32) at a time. The key instruction is that the images in the groups should be similar in every respect. Figure 7 shows two snapshots of the test. The details can be found in Ref. 32. For a typical dataset of 3000 images, ViSiProG will take 10-15 minutes to form a cluster with 9 patches. Each subject forms several groups. Then all the groups of all the subjects are analysed (with spectral clustering) to form the final clusters. For our experiment, we collected 128×128 patches from 4 1024×1024 images, and used ViSiProG to form 44 distinct clusters. Examples are shown in Figure 8.

5.2 Subjective Test II: Pairwise Similarity

In this test, the K ($K = 8$) candidates at the target matching stage of MTC are presented to the subject together with the target. To avoid unnecessary comparisons, if two candidates are point-by-point very similar to each other (based on MSE metric, which is the most conservative), we eliminate one of the two. The remaining candidates are presented to the subject, four or fewer at a time. However, the two image patches are not shown by themselves but surrounded by the corresponding image context, that is, the image surrounding the target block. Specifically, the lighting corrected candidate blocks are embedded into the context of the target block, as shown in Figure 9. This is done without using any blending, for two reasons. First, it makes the implementation simpler. Second, it is the candidate itself, rather than the blending algorithm,[33] that is critical for the evaluation of structural similarity. If anything, this approach is more conservative, forcing the subjects to select the candidates that are aligned well in both spatial domain and illumination. The size of the context

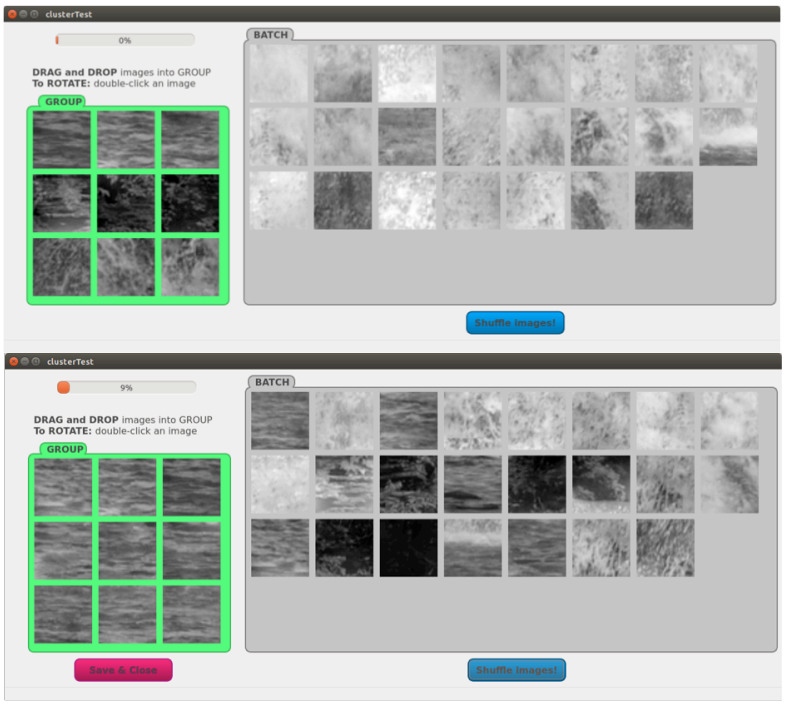

Figure 7. Two snapshots of ViSiProG

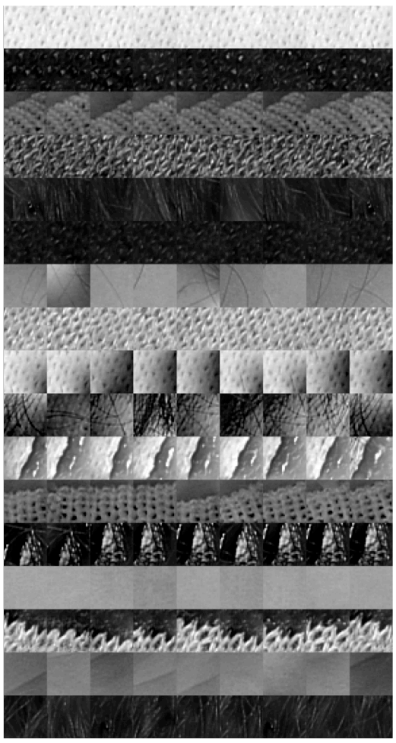

Figure 8. Examples of identical texture clusters

is selected to be 5 times the block size, if the block size is 32×32 or smaller, and 3 times the block size if the block size is bigger.

For each target block, up to four candidates are shown in one snapshot of the test. As can be seen in Figure 9, the target with its context is shown at the top left, and the candidate (or a gray square) surrounded by the same context is shown at the top right. The four candidates are shown below. When the mouse is dragged over one of the candidates, the candidate is placed in the location of the gray square, as shown in Figure 10). For each of the candidates, the subject has to select the quality of the match (highest, high, good, or bad). When all the scores are selected, the subject presses the 'Done!' button to move to the next block. If all of the candidates are bad, the subject can click the 'X' button to reject all the candidates simultaneously.

For an image with size of 1024×1024 pixels, and a 32×32 block size with at most 4 candidates to be matched, there are about 3000 pairs of patches to be scored. This is quite cumbersome, so the subject is given the option to stop at any point and to continue the test at a later time. This test is deployed on a web server whose URL could be found at the first authors' website.

6. METRIC TRAINING

We can use the database obtained from Subjective Test I to train the STSIM-I metric, that is, to select the ς_k^2 variances in (3). The database can also be used to train the STSIM-M metric, that is, to select the σ_k^2 variances in (3). We used 44 identical texture clusters for the training. Note that each cluster contains 9 identical textures as we collected data from two users only, and there was no point in do spectral clustering. Another set of 13 (different) clusters containing at total of 1174 image patches were used for testing the trained metrics. In addition, we applied n-fold ($n = 5$) cross validation with the 44 clusters. Finally, for completeness, we also tested on the training data. Table 1 summarizes the results. It compares the performance of the two metrics, STSIM-I

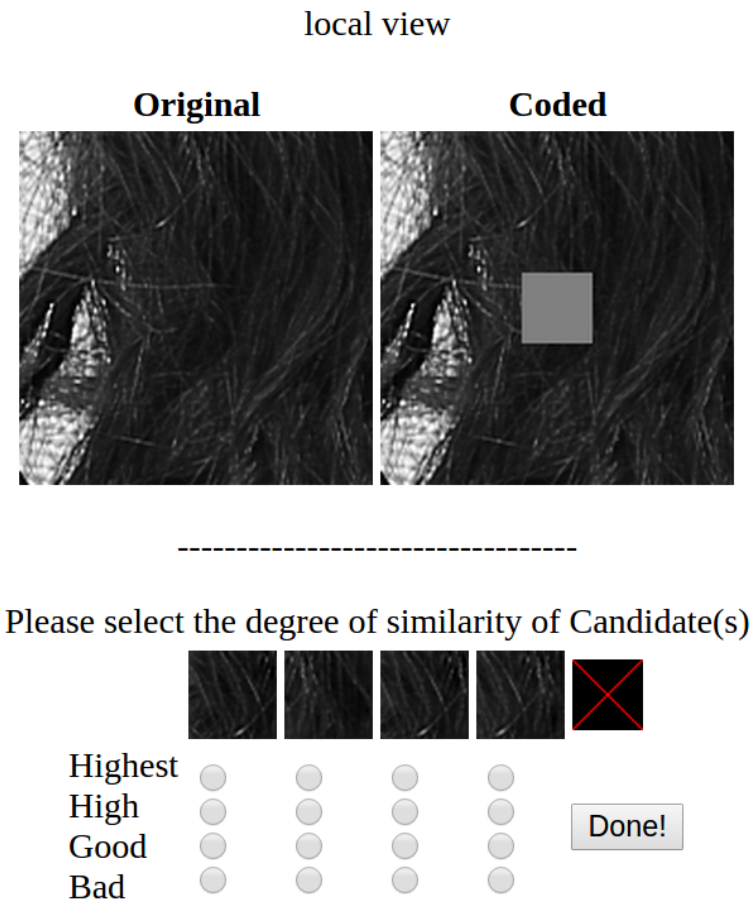

Figure 9. User Interface of MTC subjective coding mode

	Training Set	n-fold Cross Validation	Independent Set
STSIM-M	96.93%	93.50%	98.50%
STSIM-I	98.32%	95.88%	98.88%
STSIM-2	63.51%	44.00%	N/A

Table 1. Training and testing on identical texture clusters

and STSIM-M, that we trained using the database from Subjective Test I, and the STSIM-2 metric, which requires no training. Note that due to huge computation burden, STSIM-2 is not tested on the independent test set. As expected, the performance is higher within the training set than that for the n-fold cross-validation. Surprisingly, however, the results are better on the independent set. Also, as expected, the performance of STSIM-I is considerably better than that of STSIM-M. The gap is expected to increase when the data is noisy.[12]

The database obtained from Subjective Test II can be used to train the STSIM-M2 metric. The idea of Large Margin Nearest Neighbour (LMNN) algorithm can be used on the database to train a Mahalanobis matrix M such that the distance between two feature vectors of image patches $d_{i,j} = (\mathbf{w_i} - \mathbf{w_j})'\mathbf{X}(\mathbf{w_i} - \mathbf{w_j})$ is minimized when the subjective score between i and j is high or highest. In the mean time, for a target image block i, if the there is an candidate block j that has subjective score high or highest (set S) and another candidate block l that has subjective score good or bad (set D), LMNN will try to push l away with at least a gap p away from radius $d_{i,j}$:

$$d_{i,l} \geq d_{i,j} + p. \tag{4}$$

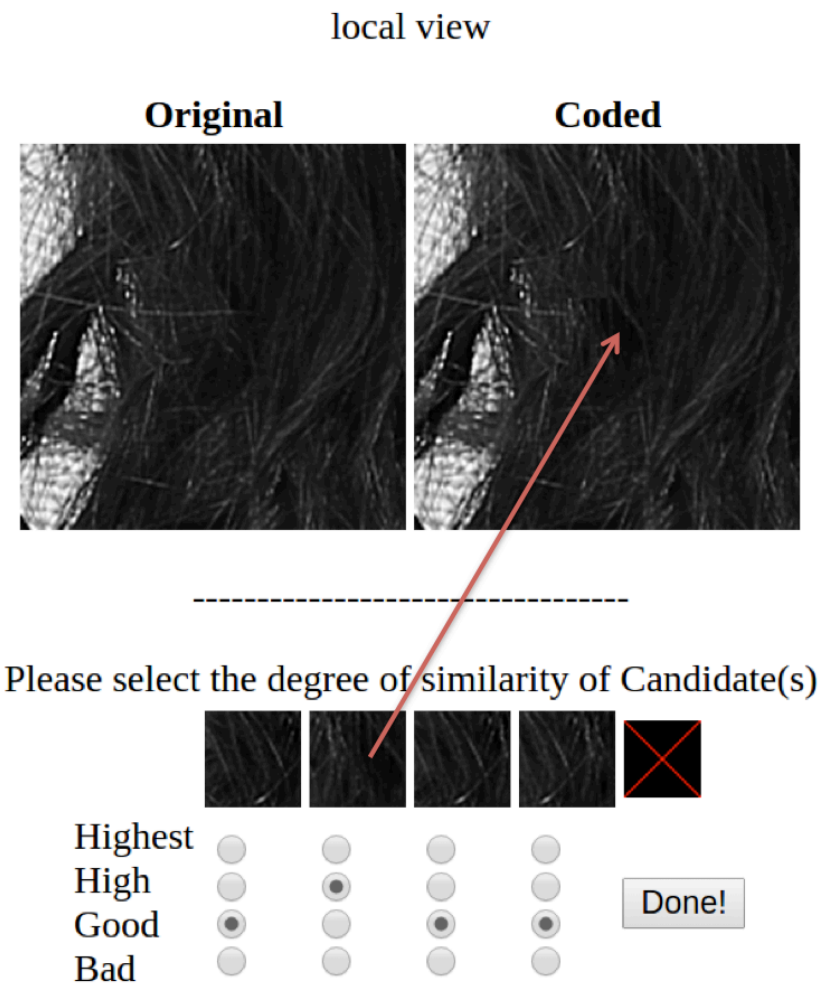

Figure 10. User Interface after subjects selection

Moreover, in structurally lossless compression, the structurally similar pairs are more important than the structurally dissimilar pairs. So a weight corresponding to subjective test score $s_{i,j}$ is also given to the distance $d_{i,j}$. The gap is given according to the relative score $p_{i,j,l} = s_{i,j} - s_{i,l}$. Thus the weighted-LMNN metric learning algorithm is:

$$\begin{aligned} &\textbf{Minimize} \quad (1-\mu) \sum_{(i,j)\in S} d_{i,j} + \mu \sum_{(i,j)\in S,(i,l)\in D} s_{i,j}\xi_{i,j,l} \\ &\textbf{subject to} \\ &\qquad d_{i,l} - d_{i,j} \geq p_{i,j,l} - \xi_{i,j,l}, \\ &\qquad \xi_{i,j,l} \geq 0, \\ &\qquad |M| \geq 0 \end{aligned} \tag{5}$$

This problem can be solved by Semi-Positive Definite Programming (SDP) with sub-gradient iteration method.

7. CONCLUSIONS

We considered the construction of databases for testing and training similarity metrics for structurally lossless image compression. Our focus was on structural texture similarity metrics and the matched-texture compression (MTC) approach. We identified "interchangeability" as a key requirement for metric performance, which is substantially more demanding than the requirements of other applications, such as image retrieval, classification, and understanding. Towards this goal, we partitioned the domain of "identical" textures into three regions, of "highest," "high," and "good" similarity. We then designed two subjective tests for data collection, the first relies on ViSiProG to build a database of "identical" clusters, and the second builds a database of image pairs with the "highest," "high," "good," and "bad" similarity labels. The databases were constructed with data generated during the MTC encoding process, and consisted of pairs of candidate and target image blocks. The identical texture clusters were used to train and test the STSIM-I and STSIM-M metrics. The context of the surrounding image was critical for the collection of the second, labelled, database. The labelled image pair database will be used in future research.

REFERENCES

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

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

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

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

[5] Jin, G., Pappas, T. N., and Neuhoff, D. L., "An adaptive lighting correction method for matched-texture coding," in [*Proc. Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP)*], 2006–2010 (May 2014).

[6] Jin, G., Pappas, T. N., and Neuhoff, D. L., "Improved side matching for matched-texture coding," in [*Proc. European Wksp. Visual Info. Proc. (EUVIP)*], (Dec. 2014).

[7] Zhao, X., Reyes, M. G., Pappas, T. N., and Neuhoff, D. L., "Structural texture similarity metrics for retrieval applications," in [*Proc. Int. Conf. Image Processing (ICIP)*], 1196–1199 (Oct. 2008).

[8] Zujovic, J., Pappas, T. N., and Neuhoff, D. L., "Structural similarity metrics for texture analysis and retrieval," in [*Proc. Int. Conf. Image Processing*], 2225–2228 (Nov. 2009).

[9] Wang, Z., Bovik, A., Sheikh, H. R., and Simoncelli, E. P., "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing* **13**(4) (2004).

[10] Wang, Z. and Simoncelli, E., "Translation insensitive image similarity in complex wavelet domain," in [*Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing ICASSP '05*], **2**, 573–576 (2005).

[11] Sampat, M. P., Wang, Z., Gupta, S., Bovik, A. C., and Markey, M. K., "Complex wavelete structural similarity: A new image similarity index," *IEEE Transactions on Image Processing* **18**(11), 2385 (2009).

[12] Maggioni, M., Jin, G., Foi, A., and Pappas, T. N., "Structural texture similarity metric based on intra-class statistics," in [*Proc. Int. Conf. Image Processing (ICIP)*], (Oct. 2014). To appear.

[13] Jayaraman, D., Mittal, A., Moorthy, A., and Bovik, A., "Objective quality assessment of multiply distorted images," in [*Signals, Systems and Computers (ASILOMAR), 2012 Conference Record of the Forty Sixth Asilomar Conference on*], 1693–1697 (Nov 2012).

[14] Ponomarenko, N. N., Ieremeiev, O., Lukin, V. V., Egiazarian, K., Jin, L., Astola, J., Vozel, B., Chehdi, K., Carli, M., Battisti, F., and Kuo, C.-C. J., "Color image database tid2013: Peculiarities and preliminary results," in [*EUVIP*], 106–111 (2013).

[15] Weinberger, K. Q., Blitzer, J., and Saul, L. K., "Distance metric learning for large margin nearest neighbor classification," in [*Advances in Neural Information Processing Systems 18*], Weiss, Y., Schölkopf, B., and Platt, J., eds., 1473–1480, MIT Press (2006).

[16] Goldberger, J., Hinton, G. E., Roweis, S. T., and Salakhutdinov, R., "Neighbourhood components analysis," in [*Advances in Neural Information Processing Systems 17*], Saul, L., Weiss, Y., and Bottou, L., eds., 513–520, MIT Press (2005).

[17] Bar-Hillel, A., Hertz, T., Shental, N., and Weinshall, D., "Learning a mahalanobis metric from equivalence constraints," *Journal of Machine Learning Research* **6**, 937–965 (Dec. 2005).

[18] Samaria, F. S. and Harter, A., "Parameterisation of a stochastic model for human face identification," in [*Proceedings of 2nd IEEE Workshop on Applications of Computer Vision*], (December 1994).

[19] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P., "Gradient-based learning applied to document recognition," *Proceedings of the IEEE* **86**, 2278–2324 (November 1998).

[20] Bache, K. and Lichman, M., "UCI machine learning repository," (2013).

[21] Dana, K., Van-Ginneken, B., Nayar, S., and Koenderink, J., "Reflectance and Texture of Real World Surfaces," *ACM Transactions on Graphics (TOG)* **18**, 1–34 (Jan 1999).

[22] T., O., T., M., M., P., J., V., and S., K. J. . H., "Outex - new framework for empirical evaluation of texture analysis algorithms.," (2002). Proc. 16th International Conference on Pattern Recognition, Quebec, Canada, 1:701 - 706.

[23] "VisTex: Vision texture database." Maintained by the Vision and Modeling group at the MIT Media Lab. web page:
`http://whitechapel.media.mit.edu/vismod/` (1995).

[24] Weinmann, M., Gall, J., and Klein, R., "Material classification based on training data synthesized using a btf database," in [*Computer Vision ECCV 2014*], Fleet, D., Pajdla, T., Schiele, B., and Tuytelaars, T., eds., *Lecture Notes in Computer Science* **8691**, 156–171, Springer International Publishing (2014).

[25] Kylberg, G., "The kylberg texture dataset v. 1.0," External report (Blue series) 35, Centre for Image Analysis, Swedish University of Agricultural Sciences and Uppsala University, Uppsala, Sweden (September 2011).

[26] Haindl, M. and Mikes, S., "Texture segmentation benchmark," in [*Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*], 1–4 (Dec 2008).

[27] Zujovic, J., Pappas, T. N., Neuhoff, D. L., van Egmond, R., and de Ridder, H., "Subjective and objective texture similarity for image compression," in [*Proc. Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP)*], 1369–1372 (Mar. 2012).

[28] Wang, Z., Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P., "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.* **13**, 600–612 (Apr. 2004).

[29] Simoncelli, E. P., Freeman, W. T., Adelson, E. H., and Heeger, D. J., "Shiftable multi-scale transforms," *IEEE Trans. Inform. Theory* **38**, 587–607 (Mar. 1992).

[30] Wiegand, T., Sullivan, G. J., Bjontegaard, G., and Luthra, A., "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Technol.* **13**, 560–576 (July 2003).

[31] Sullivan, G., Ohm, J., Han, W.-J., and Wiegand, T., "Overview of the high efficiency video coding (hevc) standard," *Circuits and Systems for Video Technology, IEEE Transactions on* **22**, 1649–1668 (Dec 2012).

[32] Zujovic, J., Pappas, T. N., Neuhoff, D. L., van Egmond, R., and de Ridder, H., "Effective and efficient subjective testing of texture similarity metrics," *Journal of the Optical Society of America A* **32**, 329–342 (Feb. 2015).

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