CONTENT BASED BLURRING CODING ARTIFACT REDUCTION USING PATCH-BASED TEXTURE SYNTHESIS

Rajas A. Sambhare and Yu Hen Hu
University of Wisconsin – Madison
Department of Electrical and Computer Engineering
Madison, WI 53706 USA
rajas@ieee.org, hu@engr.wisc.edu

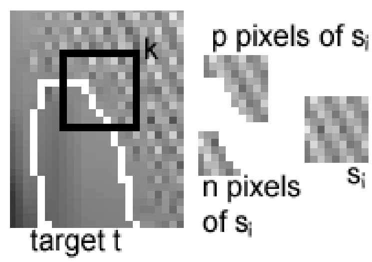
ABSTRACT

Blurring artifact occurs in low bit-rate image or video coding. It manifests itself as a blurred patch within a textured region. Previously, we proposed a constrained texture synthesis post-processing algorithm [8] to regenerate the texture the blurred patch using surrounding texture of the same kind. However, a human operator must manually identify the blurred target, and a valid source region from its surroundings. In this work, we present an effort to automate this process. Specifically, we developed an efficient modified k-means algorithm method to segment and identify potentially blurred patches; and a content-based region selection method to choose the candidate source region. Preliminary experiment results indicate that our algorithm produces results consistent with that produced by a human operator.

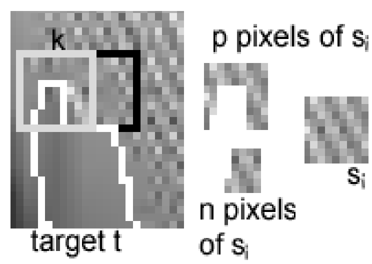
1. INTRODUCTION

Aggressive low bit rate image and video coding often leads to various coding artifacts, such as blocking, color distortion, ringing and blurring [1]. Among them, the blurring artifacts are characterized by the presence of well-preserved texture adjacent to blurred regions. Many approaches have been proposed to tackle ringing [2], [3] and blocking [4], [5] artifacts. Most of then involve smoothing the unsightly ringing and blocking. Thus these approaches are designed to remove artifacts rather than recover lost content. Previously Yang and Hu [6] have reported a dithering approach to recover some high-frequency components in a JPEG-encoded image. However, this approach seems to be applicable only to fine-grain textures like photo granular noise. Krishnamurthy et.al. have reported a texture restoration method [7] based on a Wold-decomposition. This is however applicable to a uniformly blurred image, and thus does not exploit the presence of well preserved texture in transform-coded images and videos that suffer from blurring artifacts. In [8], Hu and Sambhare demonstrate a post-processing technique based on patch-based texture synthesis that exploits the presence of well-preserved texture to patch blurred regions seamlessly. While this technique demonstrates the feasibility of such a patch based texture restoration method, that method requires a high degree of user input to identify regions suffering from blurring artifacts and potential source regions which have the same texture preserved. In this paper we develop content-based algorithm to automate this process of source and target region identification. Identifying source and target regions is essentially a texture based segmentation problem. We use Difference of Offset Gaussian filters to create a feature set and a variation of the k-means clustering algorithm to segment the image. A rule

(a) target region t and source region S are manually determined.



(b) find s_i whose p portion best match existing p portion. Use the n portion of s_i to synthesize the lost texture.



(c) The target area is now reduced by the area of the n portion that is synthesized in the last step.

Figure 1. Illustration of constrained texture synthesis procedure to reduce washout artifact.

based heuristic is then applied to the segments to identify potential source and target segments.

The rest of this paper is organized as follows. In Section 2, the patch-based texture synthesis algorithm is reviewed. Section 3 describes the segmentation technique. Section 4 describes the rule-based heuristics used for identification of source and target regions. In Section 5, we describe an experiment to quantify the success of the overall algorithm. Finally Section 6 summarizes the contributions and provides an overview of future directions.

2. BLUR ARTIFACT REDUCTION USING PATCH-BASED TEXTURE SYNTHESIS

In the previously proposed algorithm [8], refer to Figure 1, a potential target region, namely, the blurred patch, and a source region that adjacent to the target region with the same kind of texture are identified manually. From within the source region, find a small patch that can seamlessly regenerate the same kinds of texture within the blurred patch. This process is repeated until the entire blurred patch is covered with similar texture in the source region. A major drawback of this approach is the reliance on a knowledgeable human operator to identify the source and target region manually. Below, we present content-based algorithms to automate this process. Figure 2 summarizes the overall process. Note that in addition to the processing described in [8] we add a wavelet based post-processing step, which combines the high-frequency information in the patched image, and the low frequency in the degraded image, to create the final image.

3. SEGMENTATION

3.1 Feature Extraction

We represent each pixel in the image using a 13 dimensional feature vector. Features are distributed across dimensions as follows. Two features are used to represent spatial relationships between pixels; the *x* and *y* pixel coordinates normalized by image dimensions. Average color information (low pass *r*, *g* and *b*) forms 3 more features. Note that we use median filtering to generate low pass *rgb* information to preserve edge information.

The remaining 8 features are texture features. We use Difference of Gaussian (DOG) and Difference of Offset Gaussian (DOOG) filters [9] to detect spotted regions and barred regions respectively. We use two different DOG filters to detect two different spotted regions and 6 different orientations of DOOG filters. The filters are shown in Figure 3. Unlike other texture segmentation approaches we use filters at a single scale only (8 pixels wide). We do this because at higher scales, textured regions do not suffer from blurring artifacts and hence we choose not to use such regions for texture synthesis. We apply full wave rectification to the filter outputs, to model the outputs of V1 simple cells. This is followed by median filtering using a 9 pixel wide median filter, to suppress weak texture responses in the presence of strong responses (mimicking neuronal inhibition) and still maintains boundary information.

3.2 Feature dimensionality reduction

To reduce the amount of redundant information in the features, we perform Principal Component Analysis on the feature set. We retain those “important” eigenvectors for which the eigenvalues are greater than 5% of the maximum eigenvalue. We project the features into this eigenspace to get final feature set. This is the feature space accurately represents image features using the smallest number of dimensions.

3.3 Modified *k*-means segmentation

We use *k*-means clustering for segmentation. *K*-means for image segmentation suffers from the simple defect that spatially

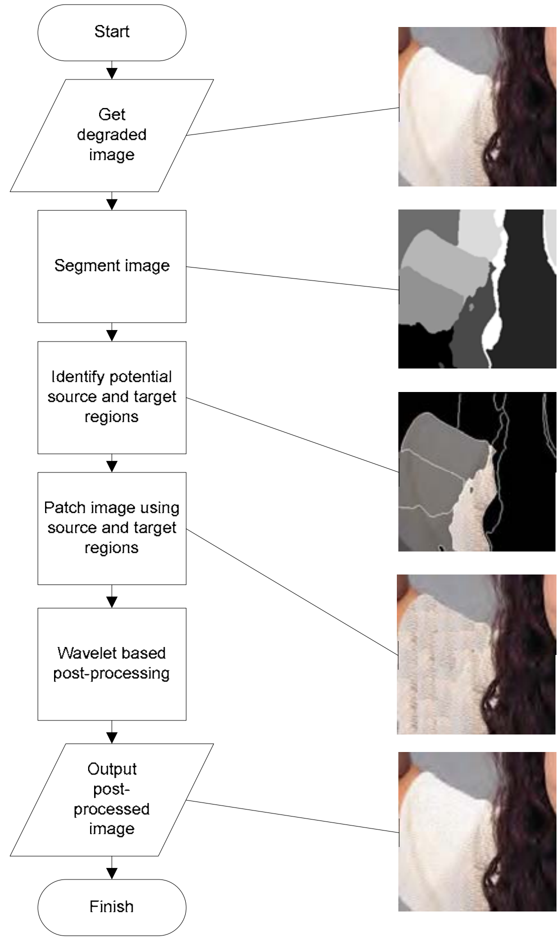


Figure 2. Block diagram of overall content based post-processing algorithm.

discontinuous regions might get classified into the same cluster. To rectify this defect we include spatial features into the feature set, as explained in the previous section. This scheme causes large regions to be split into smaller regions. However, this kind of splitting does not negatively affect the segment identification process unless it is excessive, so we continue to use *k*-means for segmentation. Note that *k*-means is an O(*cknd*) algorithm, where *k* is the number of clusters, *n* is the number of feature points, *d* is the dimensionality of the feature space and *c* is the number of iterations used which depends sub-linearly on *k*, *n* and *d*) [10]. For a simple 256 by 256 pixel image, this means that we have to classify 65536 points into say $k = 8$ clusters. Experimental run-times for this are around 5 minutes MATLAB and as *n* is proportional to the square of image dimensions, for larger images this time becomes too large. To keep the computational requirements of the segmentation stage in check we propose to modify the *k*-means algorithm as follows.

Randomly select 10% of the points of the feature space and use *k*-means to segment them into the required number of clusters. Then consider all the remaining points and use the centroids in the previous steps to classify them (assigning each point to its nearest centroid). (The key insight behind this process is that humans can segment an image quite successfully even from a randomly sampled subset of pixels.)

4. IDENTIFYING SOURCE AND TARGET REGIONS

After the image has been segmented into different barred, spotted and blurred regions, we apply a rule-based heuristic to classify regions as potential blurred artifacts and their corresponding source regions (regions where texture has not been lost). This heuristic should be characterized by low false identifications.

4.1. Classifying segments as 'textured' and 'non-textured'

As a first stage in this classification we derive a combined texture feature. To do this we apply the max operator to all texture features for each pixel. The result of this operation is thresholded using an adaptively decided threshold. To find the threshold we use the graythresh function from the MATLAB image processing toolbox. This implements Otsu's algorithm [11]. Let $|A_i|$ be the number of pixels in the ith region A_i, and let S be the textured region. Then all regions with,

$$\frac{|A_i \cap S|}{|A_i|} > 0.8$$

are considered to be 'textured'.

4.2 Rule based heuristic to classify segments as 'source' and 'target'

We construct an adjacency matrix mapping textured egments to all adjacent non-textured segments. Let $|B_i|$ be the number of pixels in the boundary of region i. Two segments i and j are considered to be adjacent when,

$$\frac{|B_i \cap B_j|}{|B_i \cup B_j|} > 0.5 .$$

We then apply the following rule-based heuristic to classify segments as source and target and update the adjacency matrix.

```
For each textured region
    For each adjacent non-textured region
        If (adjacent region is "similar" to textured region)
            Mark regions as potential source and target in adjacency
                matrix
```

The "similarity" criterion is based on the histograms of the two regions. Various criteria have been proposed for the measuring the distance between two image histograms. Puzicha et. al. have conducted an empirical evaluation of many of these criteria [12], based on which we choose the Kolmogorov-Smirnov distance. The KS distance is defined as the maximal discrepancy between cumulative distributions. If h_i and h_j are the histograms of the segments i and j (and k is the bin index), and H_i and H_j are the cumulative distributions based on those histograms, then the KS distance is defined as,

$$D(i, j) = \max_k \left|(H_i)_k - (H_j)_k\right|$$

We then find the significance of this distance to test the null hypotheses that both the histograms were drawn from the same distribution. Segments for which the probability of being from the same distribution is less then 0.4 are not considered to be potential source and target segments, and the adjacency matrix is updated to reflect this.

Table 1. Cumulative confusion matrix

Cumulative Confusion Matrix		Algorithm Classification	
		Source	Not source
Human Classification	Source	62	17
	Not source	31	199

5. EXPERIMENTS

The following experiment was conducted to test effectiveness of the proposed method. These experiments were conducted for 10 different degraded images. For each image, the segmentation-identification procedure was carried out, and the results were compared to manual identification carried out by a human observer. Multiple trials were carried out for each image. For each trial the algorithm identified all textured regions as either source regions or not source regions. A total of 309 textured regions were classified in all. A human observer then reclassified each textured region as a potential source region. A confusion matrix was thus derived for each trial, which specified number of regions which were correctly identified as source, correctly identified as not source, falsely identified as source regions (false positives) and falsely identified as not source regions (false negatives). Cumulative confusion matrices were derived and the probabilities of false alarm and miss ratio were derived. The results are summarized in Table 1.

We see that the overall performance of the algorithm is quite good with a false alarm probability of 10% and a miss-ratio of 6%. Overall, 84% of the 309 regions have been correctly categorized as valid source-target region pair, or invalid source regions. The results of applying the algorithm to a single test image are shown in Figure 3. Additional results are posted in http://mmsplab.ece.wisc.edu/~sambhare/postproc.html

6. CONCLUSION

In this work, an automated procedure is presented that segment and identify potential blur patches in a low-bit rate coded image or video, as well as a source region from which the texture can be used to synthesize the lost texture in the blurred patch. The selection is based on a Kolmogorov-smirnov distance measure evaluated on the histogram of the regions.

We observe that the results are quite good for color images, but improvements are still needed for grayscale images. This may be due to the two extra color components of the color images give more information. For JPEG2000 coded grayscale images, it is quite difficult to find boundaries between blur artifacts and naturally smooth regions manually. This problem is even more difficult to tackle automatically. One useful cue that could be used is the contour cue, where smooth contours in images could be used as boundaries between regions having similar gray levels.

Among the possible future improvements could be using a better classifier than the k-means classifier used. The recently proposed k-harmonic means classifier could be used to get better results. The normalized cuts method for image

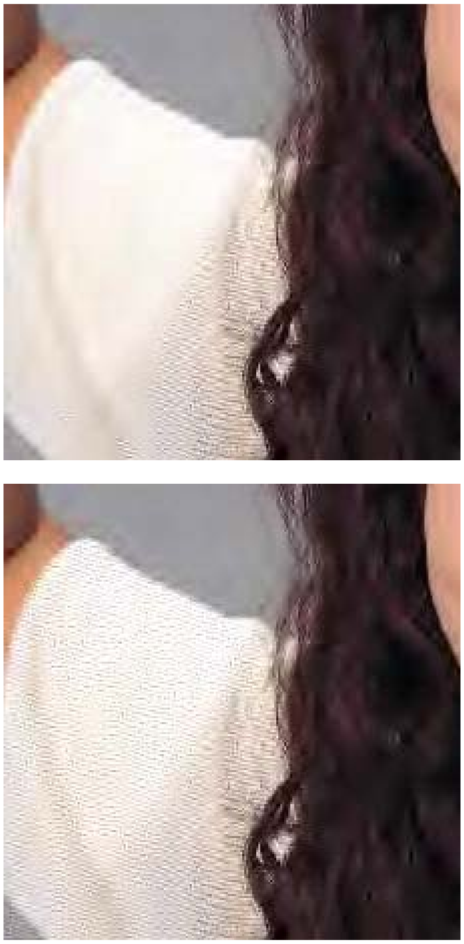


Figure 3 (a) Original degraded image with blur artifact. (b) Image post-processed with content based analysis.

segmentation is not plagued by problems like disjoint segments and splitting of large segments. While we avoided using this classifier in the paper due to its computationally intensive nature, the algorithm could be modified in a way similar to our modification of k-means by using random sampling of pixels.

7. REFERENCES

[1] Aleks Jakulin, “Baseline JPEG and JPEG2000 Artifacts Illustrated,” unpublished. Available: http://zeus.fri.uni-lj.si/~aleks/jpeg/artifacts.htm

[2] [2] S. Yang, Y. H. Hu, D. L. Tull, and T. Q. Nguyen, “Maximum likelihood parameter estimation for image ringing artifact removal,” IEEE Trans. Circuits and Systems for Video Technology, vol. 11, no. 8, August, 2001, pp. 963-973.

[3] A. Nosratinia, “Post-Processing of JPEG-2000 Images to Remove Compression Artifacts,” IEEE Signal Processing Letters, vol 10, no. 10, October 2003, pp. 296-299.

[4] S. H. Oguz, T. Q. Nguyen, and Y. H. Hu, “Critical quantization decisions in transform coding and blocking artifacts,” Proc. ISCAS'99, Orlando, FL, pp. 63.19, 1999..

[5] S. Yang, Y. H. Hu, and D. L. Tull, “Blocking artifact removal using robust statistics and line process,” IEEE Int'l Workshop on Multimedia Signal Processing, 1999.

[6] S. Yang and Y. H. Hu, “Block Effect Removal using Regularization and Dithering,” Proc. ICIP'98, Chicago, IL, pp. 346-349, 1998.

[7] R. Krishnamurthy, J. W. Wood, and J. M. Francos, “Adaptive restoration of textured images with mixed spectra,” IEEE Trans. Image Processing, vol. 5, pp. 648-652, 1996.

[8] Y. H. Hu and R. Sambhare, “Constrained Texture Synthesis for Image Post Processing”, Proc. ICASSP '03, vol 3, pp. III_281 -III_284, Hong Kong, 2003.

[9] J. Malik and P. Perona, “Preattentive texture discrimination mechanisms with early visio mechanisms”, J. Opt. Soc. Am. A. Vol 7, No. 5, May 1990.

[10] R. Duda, P. Hart, and H. Stork, “Pattern Classification,” Wiley-Interscience, New York, NY, 2 ed., 2000.

[11] N. Otsu, “A Threshold Selection Method from Gray-Level Histograms,” IEEE Transactions on Systems, Man, and Cybernetics, vol. 9, no. 1, pp. 62-66, 1979.

[12] J. Puzicha, Y. Rubner, C. Tomasi, and J. Buhmann, “Empirical evaluation of dissimilarity measures for color and texture,” Proc. IEEE International Conference on Computer Vision(ICCV-1999), pp1165-1173, 1999