LOSSY IMAGE CODING IN THE PIXEL DOMAIN USING A SPARSE STEERING KERNEL SYNTHESIS APPROACH

Ruben Verhack[⋆†]*, Andreas Krutz*[†]*, Peter Lambert*[⋆]*, Rik Van de Walle*[⋆]*, and Thomas Sikora*[†]

[⋆]Ghent University - iMinds - Multimedia Lab, Ghent, Belgium
[†]Technische Universität Berlin - Communication Systems Lab, Berlin, Germany

ABSTRACT

Kernel regression has been proven successful for image denoising, deblocking and reconstruction. These techniques lay the foundation for new image coding opportunities. In this paper, we introduce a novel compression scheme: *Sparse Steering Kernel Synthesis Coding* (SSKSC). This pre- and post-processor for JPEG performs non-uniform sampling based on the smoothness of an image, and reconstructs the missing pixels using adaptive kernel regression. At the same time, the kernel regression reduces the blocking artifacts from the JPEG coding. Crucial to this technique is that non-uniform sampling is performed while maintaining only a small overhead for signalization. Compared to JPEG, SSKSC achieves a compression gain for low bits-per-pixel regions of 50% or more for PSNR and SSIM. A PSNR gain is typically in the 0.0 - 0.5 bpp range, and an SSIM gain can mostly be achieved in the 0.0 - 1.0 bpp range.

Index Terms— image coding, compression, adaptive sampling, kernel regression, sparse steering kernel synthesis

1. INTRODUCTION

In order to cope with the ever increasing bandwidth requirements and the vast amount of imagery in the world, image coding remains an active field of study. Current image coding standards are mostly based on regularly sampled image data, which inherently means that there is a constant bandwidth for spatial frequencies over the whole images. However, this is not optimal as the constant bandwidth is likely to be excessive in some regions and too limited in other regions.

In *Content-Adaptive Coding* systems, the coding parameters are altered according to variations in signal statistics, which optimizes the system performance for non-stationary signals [1]. In the field of image processing, varying the amount of samples in a region can be done according to the amount of information inside of that region. The higher the variance in an area of an image, the more samples are taken for this region. This avoids coding superfluous samples and saves bandwidth.

In JPEG2000, the clustering of significant image coefficients along sharp edges or other features is exploited, so that for large smooth areas of the image only few coefficients are needed in their representation [2]. However, the representation of characteristic functions over smooth areas by wavelet coefficients has severe restrictions [3]. Instead of trying to represent these smooth regions by small coefficients, this paper investigates simply dropping excessive pixels to fully utilize the spatial bandwidth available.

Steered Kernel Regression (SKR) was first introduced by Takeda et al., as a general framework for kernel regression in the bivariate case [4]. The filters derived from the framework are locally adapted kernels which take into account both the local density of the available samples and the actual values of these samples. As such, they are automatically steered and adapted to both sampling geometry and the samples' radiometry, as illustrated in Fig. 1. Interestingly, this method is able to reconstruct images to an acceptable quality with no more than 15% of the original pixels [5]. Furthermore, SKR has proven to be a powerful deblocking tool [6].

Based on these two ideas, we propose a novel compression scheme called *Sparse Steering Kernel Synthesis Coding* (SSKSC). The goal is to use data-adaptive sampling and to use data-adaptive kernel regression for reconstruction. Furthermore, the signaling overhead for the sampling should be kept minimal, which is otherwise a common problem in similar approaches [7]. The scheme is implemented as a pre- and post-processor for JPEG [8]. For the reason that JPEG is still omnipresent, the fact that JPEG does not exploit any interblock correlation, and the fact that JPEG uses a small block size. The latter two reasons are particularly exploited by SSKSC.

In Section 2, the different steps in SSKSC are explained. Next, the experimental results are discussed in Section 3. Finally, the conclusions are presented in Section 4.

The research activities described in this paper were funded by Ghent University, iMinds, Communication Systems Lab - Technische Universität Berlin, the Agency for Innovation by Science and Technology in Flanders (IWT), the Fund for Scientific Research Flanders (FWO Flanders), and the European Union.

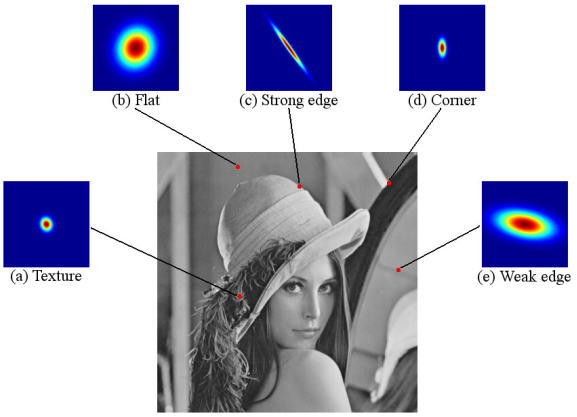


Fig. 1: Footprint examples of steering kernels. Kernels are capable of growing and shrinking according to the local sample density, and are steered along edges. (Source: [4])

2. ADAPTIVE SAMPLING AND STEERED KERNEL REGRESSION

2.1. Introduction

The idea of irregular or non-uniform sampling together with a reconstruction is not new, and all of these techniques have the same returning elements in their approaches [1][3][7][9]. Firstly, the pixels to be dropped have to be selected. Secondly, these positions need to be communicated to the decoder. Thirdly, the remaining selected pixels need to be encoded.

Particularly, it is a challenge to find an elegant way of coding the samples while preserving correlation and keeping a low overhead for the pixel locations. This signalization can take up to 50% of the bitstream in some methods [7]. In this section, we present our compression scheme that handles the aforementioned problems.

The goals for this scheme are, firstly, to drop more pixels in low-texture regions, while keeping more pixels in texturized regions. Secondly, an elegant way of passing the coordinates of the selected samples to the decoder should be found. Finally, some correlation between the selected samples should be kept, such that the selected pixels can be encoded efficiently.

The sampling approach falls into the category of *Adaptive Sampling in the Transformation Domain* [1]. For reasons mentioned above, the scheme is implemented as a pre- and post-processor for JPEG, as illustrated in Fig. 2. Finally, an adapted version of SKR was implemented exploiting extra available sample density information.

2.2. Subsampling

The first step in the process is to divide the image into blocks. Our experiments have shown that a block size of 32-by-32 pixels yields the best results.

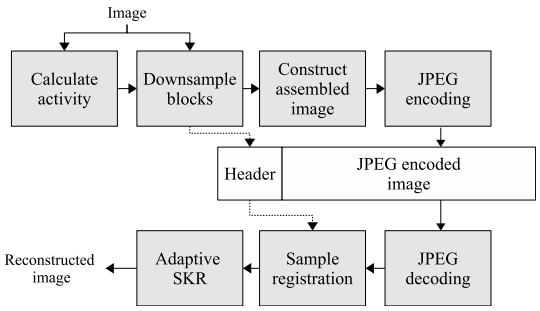


Fig. 2: A schematic representation of the SSKSC compression scheme, showing the pre- and post-processing of JPEG.

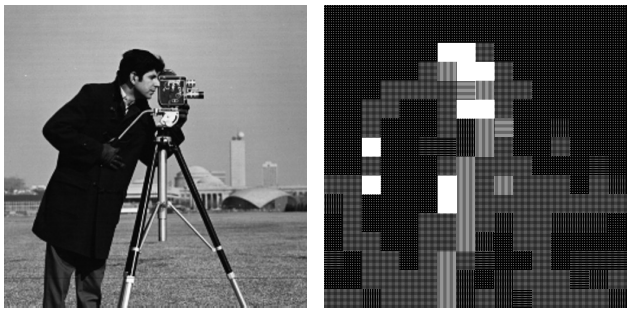
For each block, the 2D-DCT is calculated in order to calculate the vertical and horizontal *activity* inside of this block. The activity is calculated by summing the absolute values of the AC-coefficients from the 2D-DCT transformation. Ramponi et al. proposed to base the location of the samples on a measure of the *skewness* in the neighborhood of each image pixel. This method completes all the requirements and yields good results [9]. But this kind of methods are not suited for SSKSC, as our approach downsamples blockwise, in order to achieve minimal signaling overhead.

Each pixel block is downsampled horizontally and vertically according to the activity in these directions, depending on two chosen activity thresholds. If there is little vertical activity, then the block is horizontally downsampled more and analogous for vertical downsampling. In both directions downsampling is possible by a factor of 1, 2 or 4, yielding nine classes of downsampling maps. In case of downsampling horizontally and vertically by a factor of four, then 15 out of 16 pixels are being removed. Please note that the top-left pixel is kept, and the others are dropped without using any low-pass filter.

To be able to restore the downsampled blocks to their original size, a header is constructed containing the downsampling class identifier for each block. Only 4 bits per block are needed, which is very small comparing to other signaling methods [7]. An example of a sample map is shown in Fig. 3, in which white indicates a pixel that is kept.

2.3. Encoding

Now that the pixels are chosen, the pixels need to be encoded. First, the set of all different sized blocks is transformed into one assembled image, so that it can be encoded efficiently. All blocks are horizontally concatenated into one long image, starting with the classes that are downsampled the least, continuing to the classes that are downsampled the most, i.e. from blocks of class 1:1 to class 4:4. The height of the assembled image is the height of one block, which is 32 pixels. 1:1 blocks are placed next to each other, blocks that are sub-



(a) *Cameraman* (b) Sample map

Fig. 3: Example of how *Cameraman* is sampled. White indicates a pixel that is selected for encoding.

sampled vertically are stacked into (potentially incomplete) columns. This avoids that the correlation between the pixels is lost. Even though the assembled image has lost correlation between the blocks as they are arranged according to block size (instead of their spatial location), the in-block correlation is kept as the blocks have only been resized.

Secondly, the assembled image is encoded using JPEG. JPEG treats blocks independently of each other, so changing the order of those blocks does not change the coding efficiency [8]. In SSKSC, an initial block size of 32-by-32 pixels is used. Consequently, downsampling 4:4 yields the block size of JPEG. Therefore, all correlation can still be exploited by the compression algorithm.

The JPEG compression ratio of the assembled image is an important parameter in SSKSC. For example, it is possible to downsample more, but to store the image in a higher quality, or to keep more samples but compress them more.

The header of our compressed file consists of the block size, the width and height of the image, and the downsample class for every block. The total header overhead is 8 bits for the block size, 16 bits for the width and height and the number of blocks times 4 bits. This is an extremely small overhead compared to signaling the entire sampling map. For a 512-by-512 pixel image and a block size of 32-by-32 pixels, the header is 132 bytes or 0.004 bits per pixel. Finally, the encoded file then consists only of the header and the JPEG compressed assembled image.

2.4. Decoding and Sparse Steering Kernel Synthesis

The compressed file is decoded by first extracting the body, which is the JPEG encoded assembled image, of the file. Afterwards, the image is disassembled into the various sized blocks and the subsample class header is used to register all the pixels from the assembled image into the full irregularly sampled image.

Once the sparse irregularly sampled image is there, the next step is to interpolate all the missing samples. In SSKSC, interpolation is done using an adapted version of *Steered Kernel Regression* (SKR), which we refer to as ASKR [4].

Classical *kernel regression* is a non-parametric method for finding non-linear relations between pairs of random variables. This relation is an implicit model of the data [10]. As such, kernel regression provides a mechanism for computing point-wise estimates of the image with minimal assumptions on the model.

Let us consider an image as a 2D luminance function $I(\mathbf{z})$ with $\mathbf{z} = [x, y]$. $I(\mathbf{z})$ is assumed to be locally smooth to some order N, then I can be estimated at point $\mathbf{z}$ using a local expansion of the function around $\mathbf{z}$, e.g. through the Taylor series expansion [4]. The parameters of this expansion need to be estimated using a least square approach from the available samples while giving nearby samples higher weight than samples further away. To describe this distance, a 2D kernel function $K(\cdot)$ is used.

Furthermore in SKR, these kernels are *steered* (i.e. elongated, rotated, and scaled), along edges and according to the local sample density, as illustrated in Fig. 1. Consequently, pixels are interpolated from pixels with whom they share features with. Furthermore, if the image was stored using a low-quality JPEG setting, SKR reduces blocking artifacts [6].

For every pixel $\mathbf{z}$, the *steering kernel* $K_{\mathbf{H_z}}(\cdot)$ is defined by the *steering matrix* $\mathbf{H_z}$ as

$$K_{\mathbf{H_z}}(\cdot) = \frac{1}{\det(\mathbf{H_z})} K^G(\frac{\cdot}{\mathbf{H_z^{-1}}}), \tag{1}$$

where $K^G(\cdot)$ is the Gaussian kernel. The steering matrix is defined as

$$\mathbf{H_z} = h\mu_{\mathbf{z}}\mathbf{C_z}^{-1/2}, \tag{2}$$

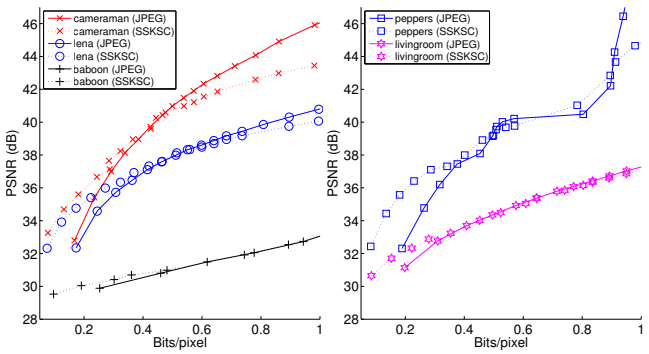
where h is the global smoothing or bandwidth parameter, $\mu_{\mathbf{z}}$ is the scalar that captures the local density, and $\mathbf{C_z}$ is the covariance matrix based on the local gray values around $\mathbf{z}$ [4]. h depends on the downsampling class of the block which contains $\mathbf{z}$: the more downsampling, the higher h should be.

In SSKSC, for each sample $\mathbf{z}$ it is known to which downsampling class q its block belongs to. If a block was downsampled 4 times in both directions (only 1/16 or 6.25% of the pixels remain), it is logical to increase the smoothing parameter h. Consequently, we let the smoothing parameter h_q be a function of q.

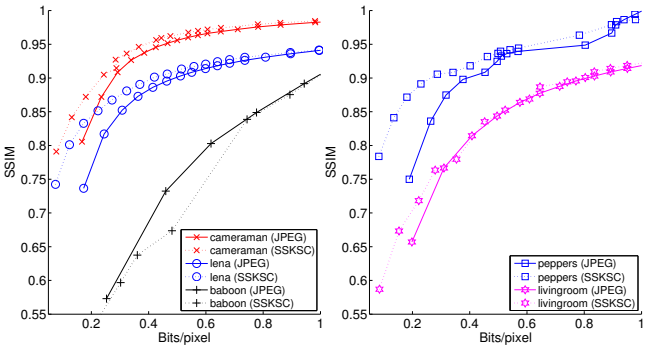
3. EXPERIMENTS AND RESULTS

The rate-distortion curves for five gray-scale images (*Baboon*, *Cameraman*, *Lena*, *Livingroom*, and *Peppers*) are shown in Fig. 4, comparing SSKSC and the standard JPEG.

The parameters, such as the encoding quality and the downsampling thresholds were trained on the *Lena* image and then applied on the four other images. The training showed that from a certain point (around 0.5 bpp) subsampling is not beneficial, but the ASKR is still deblocking the



(a) PSNR



(a) SSIM

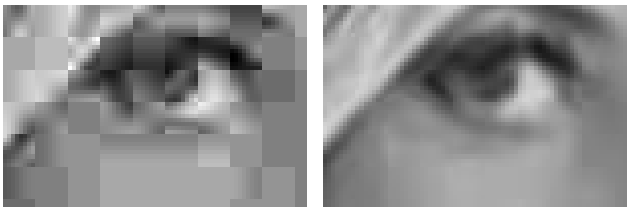
Fig. 4: Rate-distortion curves for *Baboon*, *Cameraman*, *Lena*, *Peppers*, and *Livingroom*.

image succesfully. The image quality metrics we use in this work are *Peak Signal-to-Noise Ratio* (PSNR) and *Structural Similarity Index* (SSIM) [11].

For Lena, it is evident that for the low-bpp-region (0.0 - 0.4 bpp) the new method largely improves JPEG in both SSIM and PSNR. While SSIM indicates a gain in the full 0.0 - 1.0 bpp range, PSNR only shows a gain between 0.0 and 0.5 bpp. Most interestingly, this crossing corresponds to the moment where our scheme does not subsample. Instead from 0.5 bpp onwards, only the JPEG quality varies and kernel regression is only used for deblocking. It is known that PSNR is sensitive towards deblocking, because PSNR only performs an absolute pixel-value comparison. In contrast, SSIM still shows gain until 1.0 bpp. This is because SSIM is a perceptual quality assessment, which favors more continuous structural properties. After 1.0 bpp the SKR results in unwanted blurring as JPEG does not produce any blocking artifacts.

A visual comparison between JPEG and SSKSC compression in Fig. 5 shows the strong subjective quality improvement for the same compression ratio. JPEG clearly results into a blocky image, while SSKSC results in a rather detailed image, but some blurring is visible and the eyes are less sharp than in JPEG. Nevertheless it is hard to claim that JPEG results in a visually more pleasing result.

The *Peppers*, *Cameraman*, and *Livingroom* images follow



(a) JPEG (b) SSKSC

Fig. 5: Comparing *Lena at 0.18 bpp*.

the same trend as *Lena*. Especially for *Peppers*, a huge gain can be seen in the lower bpp regions. For an SSIM Index of 0.84, the bpp is reduced by 50%. The gain is even higher for lower quality. For PSNR at 32.4 dB, the bpp is reduced by 56%. For *Cameraman*, an SSIM gain for the whole 0.0 - 1.2 bpp range can be seen, and a gain in PSNR for the first half of that range. For the lowest case, there is a bpp reduction of 52% for 0.80 on the SSIM Index. For PSNR there is roughly a 54% compression gain for the lowest quality case. *Livingroom* consists of some flat regions and quite some highly texturized regions. For PSNR we obtain a bpp reduction in the lowest quality case of 56.25 % (around 31 dB). For SSIM, a 24.5% bpp reduction is seen around 0.66. However, both gains are only for a small region, from 0.0 to 0.30 bpp.

The *Baboon* image mainly consists of fast-varying regions. The hairs of the baboon are thin and dense. In PSNR there is a large gain only in the lowest quality case where there is actually a bpp reduction of 61% for 29.5 dB. However, for SSIM the SSKSC performs worse than JPEG over the entire bpp range. A reason for this is that the structure of the hairs has changed. Nevertheless it yields a visual pleasing result. JPEG results in a heavily blocked image, although it yields a higher PSNR score.

4. CONCLUSIONS

The presented SSKSC algorithm has shown that it is beneficial to employ an adaptive, non-uniform sampling scheme for image coding. Implemented as a pre- and post-processor for JPEG, SSKSC is able to sample sparsely according to the spatial activity in the image while keeping the signaling overhead minimal and maintaining the spatial correlation for encoding. Using an adapted version of *Steered Kernel Regression*, reconstruction is possible and deblocking is performed at the same time.

From the experimental results, it is shown that a compression gain of 50% or more is often possible for PSNR and SSIM in the low quality region. For PSNR, SSKSC achieves gains for the 0.0 - 0.5 bpp range, and for SSIM, gain is seen in most images for the 0.0 - 1.0 bpp range. The compression scheme works best for images with slow varying content, rather than images with a lot of texture.

5. REFERENCES

[1] A. Habibi, "Survey of adaptive image coding techniques," *IEEE Transactions on Communications*, vol. 25, no. 11, pp. 1275–1284, 1977.

[2] A. Skodras, C. Christopoulos, and T. Ebrahimi, "The JPEG 2000 still image compression standard," *IEEE Signal Processing Magazine*, vol. 18, no. 5, pp. 36–58, 2001.

[3] L. Demaret, A. Iske, W. Khachabi, et al., "Contextual image compression from adaptive sparse data representations," *SPARS'09-Signal Processing with Adaptive Sparse Structured Representations*, 2009.

[4] H. Takeda, S. Farsiu, and P. Milanfar, "Kernel regression for image processing and reconstruction," *IEEE Transactions on Image Processing*, vol. 16, no. 2, pp. 349–366, 2007.

[5] H. Takeda, S. Farsiu, and P. Milanfar, "Robust kernel regression for restoration and reconstruction of images from sparse noisy data," *IEEE International Conference on Image Processing*, pp. 1257–1260, 2006.

[6] M. Nieves Florentìn-Nùñez, E. Lòpez-Rubio, and F. Javier Lòpez-Rubio, "Adaptive kernel regression and probabilistic self-organizing maps for JPEG image deblocking," *Neurocomputing*, 2013.

[7] H. Le Floch and C. Labit, "Irregular image subsampling and reconstruction by adaptive sampling," *IEEE International Conference on Image Processing*, vol. 3, pp. 379–382, 1996.

[8] G. K. Wallace, "The JPEG still picture compression standard," *Communications of the ACM*, vol. 34, no. 4, pp. 30–44, 1991.

[9] G. Ramponi and S. Carrato, "An adaptive irregular sampling algorithm and its application to image coding," *Image and Vision Computing*, vol. 19, no. 7, pp. 451–460, 2001.

[10] P. Wand and C. Jones, *Kernel Smoothing*, Monographs on Statistics and Applied Probability. Chapman & Hall, 1995.

[11] Z. Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.