Contour-Relaxed Superpixels

Christian Conrad[1], Matthias Mertz[1], and Rudolf Mester[1,2]

[1] VSI Lab, CS Dept., Goethe University Frankfurt, Germany
[2] Computer Vision Laboratory, ISY, Linköping University, Sweden

Abstract. We propose and evaluate a versatile scheme for image presegmentation that generates a partition of the image into a selectable number of patches (*'superpixels'*), under the constraint of obtaining maximum homogeneity of the 'texture' inside of each patch, and maximum accordance of the contours with both the image content as well as a Gibbs-Markov random field model. In contrast to current state-of-the art approaches to superpixel segmentation, 'homogeneity' does not limit itself to smooth region-internal signals and high feature value similarity between neighboring pixels, but is applicable also to highly textured scenes. The energy functional that is to be maximized for this purpose has only a very small number of design parameters, depending on the particular statistical model used for the images.

The capability of the resulting partitions to deform according to the image content can be controlled by a single parameter. We show by means of an extensive comparative experimental evaluation that the compactness-controlled *contour-relaxed superpixels* method outperforms the state-of-the art superpixel algorithms with respect to boundary recall and undersegmentation error while being faster or on a par with respect to runtime.

1 Introduction

The history of image segmentation research, when regarded on the scale of decades, exhibits clearly discernible phases during which certain method paradigms have been dominant. There has been a period of statistical models during the 1980/90ies, largely induced by the requirements and operating conditions of remote sensing and (somewhat later) image communication. Currently, in the early 2010ies, segmentation research is clearly dominated by graph-based methods, partially also variational methods. However, it is important not to mix and possibly confuse the image model (comprising the 'energy function' that should be minimal for a 'good' segmentation), vs. the optimization scheme.

We propose and extensively evaluate in the following a method for computing superpixels, that is: relatively small regions that are expected to be homogenous with respect to their internal texture. In contrast to most other currently discussed superpixel approaches, this method can be deduced from a statistical model, and homogeneity refers to 'real' texture, not the constricted interpretation of having locally smooth, or even constant, image values.

A. Heyden et al. (Eds.): EMMCVPR 2013, LNCS 8081, pp. 280–293, 2013.

Furthermore, the features used for the segmentation process can be almost arbitrarily combined from choices such as a) gray values, b) color vectors, c) texture features (= outputs of texture operators applied onto a spatial neighborhood centered on the regarded pixel), d) depth values (e.g. from an active depth camera) e) motion vectors, and many more. Thus we speak of a whole family of segmentation modules that can be constructed from the basic approach. The only constraint is that the individual feature channels can be regarded as uncorrelated – for reasons that become clear later (section 4.1). The two main contributions of the paper are a statistically sound approach to obtain a superpixel energy function, allowing explicit control of compactness, and an extensive comparative evaluation with state-of-the art approaches for a) the base line gray value version of *contour-relaxed superpixels* and b) the color version.

2 Related Work

The term 'superpixels' dates way back in the 1980ies and has been popularized by Ren and Malik [1]. It can be defined as a connected group of pixels similar with respect to certain features, i.e. their color value. A superpixel segmentation of an image is an oversegmentation into typically many subregions, all of them expected to be a proper subset of exactly one 'semantic' region. Usually, superpixel segmentations serve as precursor to higher level tasks such as object segmentation, motion estimation or tracking [1–4] where computations are more efficiently done on some few hundred groups of such pixels [1, 3, 5] instead of $10^5 - 10^6$ pixels.

Superpixels can be obtained using standard segmentation algorithms, e.g. the well known mean-shift [6] and watershed [7] algorithms. However, it has been shown that these 'general purpose' algorithms typically produce superpixels of highly irregular shape and size and do not allow to directly control the number of superpixels [8, 9]. It is a common view in the literature [8–10] that superpixel algorithms should include a compactness constraint and should allow to directly control the number of superpixels (to avoid a costly parameter search) such that superpixels are similar in size. While non-compact superpixels can adapt to quite complicated object shapes, they risk a larger extent of *undersegmentation error*, a measure of overlap between a single superpixel and multiple objects. In our approach, the degree of compactness can be controlled by a single parameter. Variation of this parameter allows a system designer to chose between highly compact superpixels and an almost 'fluid' behavior that aligns superpixels well even with complicated boundaries.

The various approaches proposed since [1] can roughly be grouped into *graph-based* and *gradient-based* methods. Among the most popular graph-based methods one finds the superpixel algorithms based on normalized cuts [1, 5, 11] and Felzenszwalb & Huttenlocher's approach [12] who compute a superpixel segmentation by solving a shortest spanning tree problem. A drawback of [12] is that they do not encode a compactness constraint. A problem of the normalized cut approach [1] is its computational effort, since (depending on the image size) a

single run of the algorithm may take minutes on hardware as of 2012. In the 'superpixel lattices' proposal [13], superpixels are forced to conform to a grid by introducing a topology constraint leading to superpixels with a rectangular shape. SP-Lattice is among the fastest superpixel methods up to now [13, 14], but produces results of lower quality compared to more recent work [8, 9, 14].

More recently, Veksler et al. [9] and [14] computed superpixel segmentations within the well known *graph cuts* and *expansion moves* framework. Veksler et al. cover an image with overlapping square patches of fixed size which are subsequently stitched during optimization using $\alpha-$expansion. They define a second order energy function containing a data term modeling the likelihood for a specific label for a single pixel and a prior term resembling the Potts model. Two different flavors of the approach are proposed, namely 'compact' and 'constant' superpixels, respectively. The 'compact' superpixels have the inherent problem that the likelihood term does not differentiate between pixels having different color, but assigns low energy to all labels (patches) that overlap the regarded pixel. Therefore neighboring pixels are likely to be assigned the same label regardless of their color, leading to superpixels possibly containing strong discontinuities. In 'constant superpixels', a different data term is used which is based on the distance in color space between the regarded pixel and the color of the center pixel of the current label thus encouraging superpixels of constant or similar intensity. For more than 400 superpixels, the constant version of [9] outperforms TurboPixels, the normalized cut approach as well as the method by Felzenszwalb & Huttenlocher. However, while Veksler et al. encode a compactness constraint, their method does not allow to control the number of superpixels directly but would need a parameter search to do so. The work of Zhang et al. [14] relies on pseudo-boolean optimization and is inspired by the one by Veksler et al., trying to overcome the aforementioned difficulties by design. They report runtime figures much lower than the ones by Veksler et al. which are independent of the number of superpixels. However the quality of their superpixels with respect to boundary recall R_B and undersegmentation error E_{us} is worse than Veksler's.

TurboPixels [8], and more recently SLIC superpixels [10] are prominent gradient based methods. Such methods do not formulate segmentation as some graph based related problem such as *mincut* but rather optimize the energy function in a gradient ascent/descent sense. TurboPixels are based on the geometric flow implemented via level sets. The method allows to directly control the number of superpixels and integrates a compactness constraint. In the SLIC ('simple linear iterative clustering') method [10], the computation of superpixels is cast as a clustering problem in a five dimensional feature space consisting of the three `Lab` color channels and the pixel 2d coordinates. The scheme starts with regularly sampling N cluster centers in image space, subsequently perturbing the center locations such that they lie at the lowest image gradient position within a small sub window. Then pixels are assigned to the best fitting cluster center within a small neighborhood and cluster centers are recomputed based on the ℓ_1 norm between old and new center positions. The method can directly control the number of superpixels and encodes a compactness constraint. It is shown [10] that SLIC

outperforms TurboPixels, the method of Felzenszwalb & Huttenlocher, and the normalized cut approach of [1, 5].

3 Outline of the New Approach

In this work we propose a superpixel algorithm that has the following advantages: (a) direct control of the number of superpixels, (b) control of the compactness of the superpixels by setting a (single) compactness parameter κ, thus superpixels can be allowed to adapt to complicated shapes if needed, and (c) most importantly: an explicit statistical modeling of superpixel shape and content allows to perform the segmentation on an arbitrary number of feature channels, e.g. intensity-only, color, depth, or 'real' texture feature vectors. The energy function to be optimized is derived from this statistical image model, and it turns out that already a local optimization is sufficient for yielding results competitive or superior to the state of the art. The image model used for *contour-relaxed superpixels* is based on homogeneously textured regions; this includes the plain model of smooth (=quasi-constant) gray or color values, but goes significantly beyond this.

4 Theory of *Contour-Relaxed Superpixels*

In this section, we summarize the theoretical basis on which the *contour-relaxed superpixels* approach is built. Attempts to formulate the segmentation problem as an estimation task can already be found in the very early literature. Still, the potentials and advantages of a well-founded statistical model for low-level segmentation are not reflected in the current literature. We build on the fundamental model used by Mester et al. [15, 16] and transform it into a competitive superpixel approach by introducing a compactness term.

Let the total set of measurements on the image array (= a 2D-array of vectors), be summarized in a single huge vector $\boldsymbol{z}$. Let $\mathcal{Q} = \{R_1, R_2, \ldots R_n\}$ be a partition of the image array. The measurements *inside* of each region are the outcomes of region-specific stochastic processes, and each such process is associated with an individual parameter vector $\boldsymbol{\theta}_i = \boldsymbol{\theta}(R_i)$. The combination of a partition $\mathcal{Q}$ and the parameter ensemble $\{\boldsymbol{\theta}\} = \{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots \boldsymbol{\theta}_n\}$ is denoted as the 'array state' $\mathcal{S}$. Each completely specified array state $\mathcal{S}$ induces a joint probability density $p(\boldsymbol{z}, \mathcal{Q}, \{\boldsymbol{\theta}\}) = p(\boldsymbol{z}, \mathcal{S})$ for the ensemble of random variables $\boldsymbol{z}$. Given an image measurement vector $\boldsymbol{z}$, the notion 'segmentation' means to find an array state $\mathcal{S}$ which has a high likelihood to have generated the observed image vector $\boldsymbol{z}$.

4.1 Deriving the Segmentation 'Energy Function' from Maximum-A-Posteriori (MAP) Principles

The particular combination of a partition $\mathcal{Q}$ and the corresponding model parameters $\{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots \boldsymbol{\theta}_n\}$ that maximizes the probability density function

$$p(\mathcal{S}|\boldsymbol{z}) = p(\mathcal{Q}\ ,\ \{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots \boldsymbol{\theta}_n\}|\boldsymbol{z}) \tag{1}$$

is considered as the maximum-a-posteriori (MAP) estimate of the array state $\mathcal{S}$. From Bayes' theorem we obtain

$$p(\mathcal{S}|\boldsymbol{z}) = p(\boldsymbol{z}|\mathcal{S}) \cdot p(\mathcal{S})/p(\boldsymbol{z}). \tag{2}$$

With the observed image vector $\boldsymbol{z}$ being fixed, $p(\boldsymbol{z})$ is merely a normalizer. Thus we search the particular array state $\mathcal{S}$ which maximizes the target function J

$$J := \quad p(\boldsymbol{z}|\mathcal{S}) \cdot p(\mathcal{S}) = p(\boldsymbol{z},\ \mathcal{S}) = p(\boldsymbol{z}|\mathcal{Q}\,, \boldsymbol{\theta}) \cdot p(\boldsymbol{\theta}|\mathcal{Q}) \cdot p(\mathcal{Q}). \tag{3}$$

The joint density in Eq. 3 consist of the prior probability $p(\mathcal{Q})$ for the regarded partition $\mathcal{Q}$, and the conditional densities for $\boldsymbol{z}$ given the individual region model processes. Using a Gibbs random field (GRF) with discrete two-element cliques, $p(\mathcal{Q})$ is expressed as

$$p(\mathcal{Q}) = \frac{1}{Z} \cdot \exp\left(-\sum_{c_i} V_c(c_i) \right). \tag{4}$$

Here, c_i denote all maximal cliques of size 2. The potentials $V_c(c_i)$ depend on whether the label values in such a clique are identical or not (Potts model). While computing the partition function Z is intractable in general, it is not needed here, as we are only interested in the MAP estimate of Eq. 3.

The region-specific model parameter vectors $\boldsymbol{\theta}$ are considered as unknown deterministic parameters, (i.e. we assume a 'flat', uninformative prior for them) such that only the distribution for the partition $\mathcal{Q}$ appears in $p(\mathcal{S})$

$$p(\mathcal{S}) = p(\boldsymbol{\theta}, \mathcal{Q}) = \alpha \cdot p(\mathcal{Q}). \tag{5}$$

The model parameters $\boldsymbol{\theta}$ are obtained by maximizing the term $p(\boldsymbol{z}|\mathcal{S})$ with respect to the parameter $\boldsymbol{\theta}$ while the partition $\mathcal{Q}$ is fixed, that means by a region-specific maximum likelihood estimation ('EM style'):

$$\begin{aligned} p(\boldsymbol{z}, \mathcal{S}) &= p(\boldsymbol{z}|\mathcal{S}) \cdot p(\mathcal{S}) \\ &= p(\boldsymbol{z}|\mathcal{Q}\,, \{\hat{\boldsymbol{\theta}}^{ML}(\mathcal{Q})\}) \cdot p(\mathcal{Q}) \cdot \alpha \end{aligned}$$

The texture processes of the individual regions are considered to be pairwise statistically independent between regions. This means that knowing the complete texture signal inside region R_i does not yield *any* information on the texture signal inside region $R_j, j \neq i$. Thus, the joint probability density of observing *all* the texture signals (= the complete vector-valued image $\boldsymbol{z}$) can be written as a product, bearing in mind that this is an approximation (due to the ML parameter estimate):

$$p(\boldsymbol{z}|\mathcal{Q}) = \prod_{R_i} \prod_{k} p(\boldsymbol{z}_{ik}|\boldsymbol{\theta}_i^{ML}) \tag{6}$$

Here i denotes the region, and index k varies over the feature channels. This plain double product is of course the result of the independence between regions, on one hand, and the (assumed) independence between feature channels, on the other hand, which may be a coarse but effective approximation in some cases.

4.2 The Optimization: 'Contour Relaxation'

Given an initial partition Q_0, J in Eq. 3 is maximised by variation of pixel labels. Each grid point x_0 which is located on the contour of a region is regarded and it is checked whether a change of its initial region label into a another label occuring in its neighborhood leads to an increase of target function J (Eq. 3). If this is the case, the change of the label is carried out. The focusing on *contour pixels* and a *local subset* of labels is essential for the speed and explains how the method is related to the general ICM framework [17].

Due to the conditional independence structure induced by a pairwise GRF, only the 8 cliques including site x_0 need to be taken into account. Hence the expression $p(\mathcal{Q})$ (Eq. 4) can be factorized into

$$p(\mathcal{Q}) = k_1 \cdot \exp(-n'_B B - n'_C C), \tag{7}$$

where only the second factor depends on the label of x_0. Here, n'_B and n'_C denote the numbers of inhomogeneous horizontal/vertical and diagonal cliques where B and C are the associated costs, respectively.

Given the partition $\mathcal{Q}$, the conditional likelihood of the image data $\boldsymbol{z}$ can be factorized into a constant and a variable term:

$$p(\boldsymbol{z} \mid \mathcal{Q}) = k_2 \cdot \prod_{\{\mathcal{R}_j\}} p\big(\boldsymbol{z}(\mathcal{R}_j) \mid \boldsymbol{\theta}(\mathcal{R}_j)\big), \tag{8}$$

with the variable product here comprising only those regions R_j that may include pixel x_0. From the set of legal choices of $q(x_0)$, the label maximizing

$$p(\boldsymbol{z}, \mathcal{Q}) = k_1 \cdot k_2 \cdot \exp(-n'_B B - n'_C C) \cdot \prod_{\{\mathcal{R}_j\}} p\big(\boldsymbol{z}(\mathcal{R}_j) \mid \boldsymbol{\theta}(\mathcal{R}_j)\big) \tag{9}$$

is then assigned to point x_0. Due to the extreme value range of (9), it is computationally mandatory to minimize the negative log of (9) instead of directly maximizing it. This yields the main part of the *energy function* L to be minimized in the *contour-relaxed superpixels* framework (see sec. 5.1).

This scheme denoted as 'contour relaxation' is performed by scanning the whole image using the 'coding scheme' proposed by Besag [18] to avoid directional preferences. The computational expense is rather low, as only pixels on the region boundaries are considered during optimization. The parameter values B and C of the Gibbs model are far from being critical. See section 6 for the values used in our experiments, and the number of passes over the image array.

5 Specific Design Features of the Proposed Approach

The combination of a probabilistic target function to be maximized (or minimized, if the negative logarithm is regarded) and a greedy iterative optimization scheme, as described in the preceding section, leaves of course ample space for

selecting the features, the shape of the assumed distributions (Gaussian, Laplacian, etc.) in the feature channels. The number of channels, and the kind of information assigned to the channels (gray values, color components, depth values, texture features, ...) allows many variants of the fundamental scheme, yielding a whole *family* of segmentation methods. What is *not* needed are weight factors that weigh the relative importance of the individual features; since the moments of the distributions are estimated in the course of the process, the correct balance between the features is obtained automatically — this is certainly one of the main advantages of the approach.

What remains to be explained is how the functional form of the feature pdfs, the likelihood terms appearing in the energy function, and the actual data in the regions are tied together. This is done by simply computing *unnormalized moments* of the data $f_k(\boldsymbol{x}_i)$ in each region R_j, that is:

$$N := \sum_{i \in R_j} 1 \qquad S := \sum_{i \in R_j} f_k(\boldsymbol{x}_i) \qquad Q := \sum_{i \in R_j} f_k^2(\boldsymbol{x}_i) \tag{10}$$

From these *sufficient statistics* N, S, and Q, the ML estimates of the pdf parameters in channel k can be computed, and thus the likelihood for a given ensemble of data values can be determined.

5.1 Adding a Compactness Term κ

Experiments with the original model from [15, 16] show that using this universal segmentation scheme for the particular purpose of computing superpixels suffers from the compactness of the regions not being directly controllable. We introduce an additional compactness term that ensures that the (fundamentally fluid) regions do not create too wriggled, too elongated regions, as this could be the case for a pure Gibbs-Markov random field model with realistic parameters.

Carrying over the cost functional L from the MAP criterion in section 4, we can now enforce spatial compactness for the regions to be formed, and define a new cost functional $\tilde{L}$ by adding an extra 'regularization' term which penalizes the squared deviation between the spatial location $\boldsymbol{x}$ of the pixels in the region R_j and the center $\boldsymbol{m}$ of the region R_j, as follows

$$\tilde{L} = L + \kappa \cdot \sum_{\boldsymbol{x} \in R_j} (\boldsymbol{x} - \boldsymbol{m}(R_j))^T (\boldsymbol{x} - \boldsymbol{m}(R_j)), \tag{11}$$

where κ is a parameter which controls the compactness of the regions. It is straightforward to show that $\tilde{L}$ can be fully expressed in terms of these 'spatial moments' introduced in Eq. 11, the ensemble of clique potentials, and the sufficient statistics N, S, and Q previously defined, if the *functional form* of the feature distributions is given (e.g. a Gaussian model). In case that a Laplacian pdf is chosen, a corresponding set of sufficient statistics can likewise be defined, by incorporating the sum of the absolute values of the scalar features instead of squares. Likewise, other distributions such as χ^2 or Rayleigh pdfs can be used.

6 Evaluation and Experimental Results

The *Contour-relaxed superpixels*[1] approach has been evaluated for many of the numerous variations comprised by the approach; we present in the following the most relevant test results as permitted space allows. The quality of the method is quantitatively expressed by the two established measures *'boundary recall'* (R_B) and *'undersegmentation error'* (E_{us}) evaluated on the 300 images from the *Berkeley Segmentation Database* (BSDS300) [19] which contains ground truth segmentations provided by human subjects. The boundary recall R_B is a measure of how well the superpixel boundaries align with ground truth segments, while the undersegmentation error E_{us} measures the degree of bleeding caused by superpixels overlapping more than one ground truth segment. We compute the undersegmentation error E_{us} using the definition of [8] and the boundary recall using the MATLAB code provided by the BSD benchmark.

The experiments have been carried out based on a straightforward single threaded and unoptimized C/C++ implementation. For images available in the Berkeley database with a resolution of 481×321 (or 321×481) pixels, one iteration of the contour relaxation on a single channel takes about 40 ms on an Intel Xenon 2.8GHz processor.

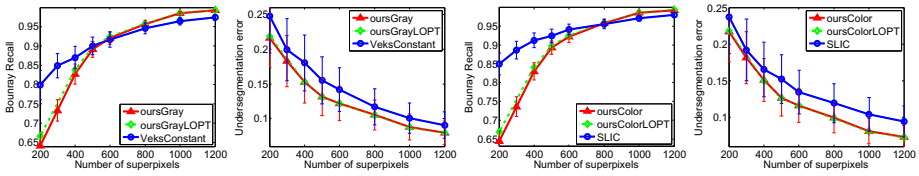

Fig. 1. Column 1 and 2: Benchmark results using intensity and compactness feature: Average boundary recall, and undersegmentation error including error bars ($\pm\sigma/2$) on the BSD300 dataset (300 images). **Column 3 and 4:** Benchmark results using color and compactness feature. Best viewed in color.

As the BSD benchmark differentiates between gray and color image segmentation, we divide our evaluation in two parts, superpixels computed on gray value images and color images, respectively. We compare our results with the state-of-the-art superpixel methods for both settings, namely the approach by Veksler et al. [9] and SLIC superpixels [10], where both of them also include a compactness term. Note that there are many other algorithms available, e.g. Turbo-Pixels [8], NCuts [1], Lattice Superpixels [13] and recently QPBO superpixels [14]. However, it has been shown [10, 14] that SLIC and the approach by Veksler et al. outperform the aforementioned algorithms with respect to boundary recall and undersegmentation error but are coequal or better with respect to runtime performance. Therefore we will exclude these methods from the qualitative evaluation but will reference them within the discussion of the runtime

[1] Code will soon be available at `http://www.vsi.cs.uni-frankfurt.de`

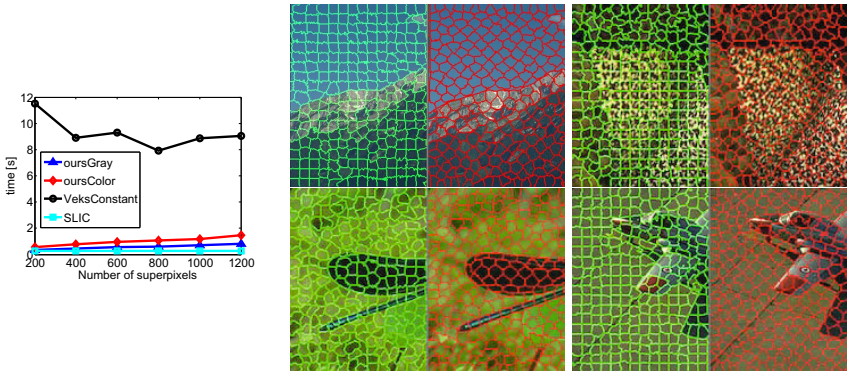

Fig. 2. (left) Runtime performance: Average runtime in seconds averaged over the BSD300 dataset (300 images). **(right)** Typical results obtained with our method when initialized with axis aligned square blocks (green boundaries), and when initialized with a diamond pattern (red boundaries).

performance. Note that we compare our superpixel approach to its most serious competitors but do not include full (object based) segmentation algorithms, e.g. gPb-owt-ucm and its variants [20], as this would not be a fair comparison. However, our approach could be used as an integral part of such a full segmentation.

Single Channel Superpixels. Within the following experiments, we configure the algorithm to use a single gray value feature channel and the compactness term. We control the number of generated superpixels by initializing the label array with a 'blind' segmentation consisting of equally-sized square blocks. Following this initialization, we perform twelve passes of the contour relaxation. The clique costs for inhomogenous horizontal or vertical and diagonal cliques were set to $B = 0.3$ and $C = \frac{\sqrt{0.3}}{\sqrt{2}}$, respectively, The value of the compactness parameter κ was 0.015. These parameters have been kept constant for all the experiments. We compute the boundary recall and undersegmentation error for 200, 300, 400, 500, 600, 800, 1000 and 1200 superpixels by averaging over these measures for the 300 images from the BSD. We compare our results with the approach by Veklser et al. [9] using their publicly available implementation[2] while setting the parameters to the ones reported in their paper. Note that it is not possible to directly control the number of superpixels generated by their algorithm but indirectly with a patch size parameter. We set the patch size parameter such that on average the method computes, say, 500 superpixels while the exact number of superpixels produced can vary for each individual image. Figure 1 shows the boundary recall and undersegmentation error for the proposed method (`oursGray`) and the one by Veksler et al., specifically the 'constant' version (`vekslerConst`) (cf. Sec. 2). Note that Veksler et al. use a global optimization method (graph cuts), while we achieve our results merely with local greedy optimization. In order to allow an evaluation of the influence

[2] `http://www.csd.uwo.ca/faculty/olga/Code/`

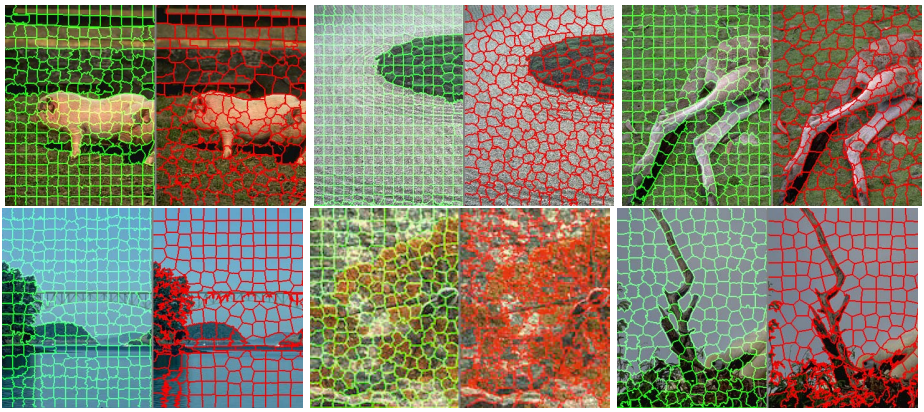

Fig. 3. *Contour-relaxed superpixels* results on a set of test images from the Berkeley database. The green boundaries show the superpixel results produced by our method compared to (top row) Veksler et al. in red when initialized with 500 (column 1 and 3) and 600 (column 2) superpixels and (bottom row) to SLIC in red when initialized with 400 (column 1 and 2) and 500 (column 3) superpixels. Best viewed in color.

of the number of iterations during optimization, we additionally compute the R_B and E_{us} measures while letting our method perform as many runs of the contour relaxation until no label changes occur, that is until a local optimum is reached (`oursGrayLOPT` in Fig. 1). From Fig. 1 it can be seen that while `vekslerConst` achieves a better boundary recall when the desired number of superpixels is relatively small (200 to 400), our *contour-relaxed superpixels* method performs better in terms of boundary recall for 550 or more superpixels. Furthermore, our method consistently outperforms its competitor `vekslerConst` with respect to undersegmentation error. Figure 1 also shows that the actual number of iterations of the contour relaxation influences the results only marginally. While more iterations slightly improve the boundary recall for 200 to 400 superpixels, they nearly have no influence on the undersegmentation error and are non-critical in practice. Figure 2 (left) shows graphs of the average running time over 200 to 1200 superpixels. While `vekslerConst` roughly takes 7 to 12 seconds, our method only takes 0.3 to 0.7 seconds. Note that the runtime of our method slightly increases with the number of superpixels: as more superpixels lead to more contour pixels in the label array, more pixels have to be visited and evaluated for a possible label change. Our method also compares favorably to Lattice- and QPBO superpixels where the authors of [14] report an average runtime for QPBO superpixels (independent of the number of superpixels) of 0.5 seconds, but are outperformed by the Veksler et al. approach with respect to boundary recall and undersegmentation error. Furthermore, [14] report that their method is usually 10% slower than Lattice superpixels. Our method has a comparable runtime while outperforming both methods in terms of quality, as expressed by R_B and E_{us}.

Multi-channel Superpixels. Within the following experiments we set up our algorithm using three independent feature channels one for each of the three YUV color channels and the compactness term. We use the same label array initialization as for the single feature case and only rescale the compactness parameter κ to 0.045 to account for the increased number of feature channels. Again, these parameters have been kept fixed while computing the boundary recall and undersegmentation error for 200, 300, 400, 500, 600, 800, 1000 and 1200 superpixels by averaging over these measures for the 300 images from the BSD. We compare our results with SLIC superpixels using their publicly available implementation[3] where the exact number of superpixels can be controlled. As for the approach by Veksler et al., we set the parameters of the SLIC implementation to the ones reported by the authors in the corresponding paper. Figure 1 shows the boundary recall and undersegmentation error for the proposed method (`oursColor` and `oursColorLOPT`) and SLIC (`SLIC`). It can be seen that we consistently perform better than `SLIC` with respect to undersegmentation error even for few superpixels. Note that the margin by which we outperform `SLIC` increases with the number of superpixels. For 200 to 750 superpixels `SLIC` achieves a better boundary recall, however, our method is coequal or better than `SLIC` for 750 or more superpixels. As for the single feature case, the actual number of iterations of contour relaxation only has a marginal effect on the results as can be seen from the boundary recall and undersegmentation error for `oursColorLOPT` compared to `oursColor`. Figure 2 (left) shows graphs of the average running time over 200 to 1200 superpixels. Here, `SLIC` takes 0.25 seconds independent of the number of superpixels while our method needs 0.6 to 1.7 seconds depending on the number of superpixels. Note that our method can easily be parallelized, that is by computing the region statistics for each channel in parallel thus making it possible to achieve a similar runtime as for the single channel version with 0.3 to 0.7 seconds. Furthermore our runtime performance also compares favorably over TurboPixels and NCuts which take several seconds and more than 30 seconds (both depending on the number of superpixels, see [8]), respectively. For a qualitative evaluation, Fig. 3 shows example superpixel segmentations using the proposed *contour-relaxed superpixels* approach, the method of Veksler et al. and SLIC. One can see that the obtained boundary maps respect the prominent boundaries of the image, and furthermore, the effect of the compactness term is clearly visible, as the resulting regions are compact and regularly shaped. As can be seen in Fig. 3, the *contour-relaxed superpixels* approach considers the strongly textured areas as homogeneous and textured, and does not spend 'energy' on complicated region contours, in contrast to Veksler et al. and SLIC shown besides. Furthermore, Fig. 2 (right) shows that the initial segmentation only has a minor influence on the results and that our approach does not have an intrinsic bias towards a specific boundary layout (horizontal/vertical). While in areas of homogeneous texture the segmentation stays close to the initial partition, superpixels adapt to shapes as necessary.

[3] `http://ivrg.epfl.ch/supplementary_material/RK_SLICSuperpixels/index.html`

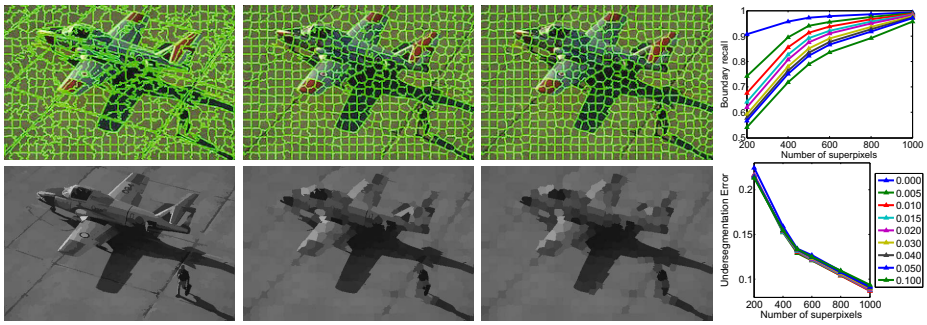

Fig. 4. Influence of varying the compactness weight: Top row, columns 1-3: *contour-relaxed superpixels* results for the 'plane' image when setting the compactness weight to 0, 0.015 and 0.03, respectively. Bottom row, columns 1-3: Region mean images corresponding to the superpixel segmentations in the top row. Column 4 shows the boundary recall and undersegmentation error over the number of superpixels averaged over all images from the BSDS300 while varying the compactness weight between 0 and 0.1. Best viewed in color.

Increasing Boundary Recall by Lowering the Compactness Weight. The precision with which the boundaries are recovered can additionally be improved, leading to a higher boundary recall by decreasing the weight of the compactness parameter κ whereas the number of relaxation passes (here: 12) only has a minor influence as shown previously. Figure 4 shows several superpixel segmentations and corresponding region mean images for the 'plane' image from the BSD visualizing the effect of lowering the compactness weight on the resulting superpixels. When the compactness term is set to 0 we can achieve a boundary recall of around 0.9 for only 200 superpixels as shown in the recall graph in Fig. 4. The spread of a superpixel is then only governed by the inhomogeneous clique potentials. In contrast to the common view in the literature [8–10] that superpixel algorithms should include a compactness term in order to avoid bleeding effects, the non-compact version of our method is only marginally influenced with respect to the undersegmentation error as shown in the graph in Fig. 4, as due to our model the superpixels still respect object boundaries. However, in this setting superpixels have a highly irregular shape which might be suboptimal for different applications. Furthermore Fig. 4 shows that when increasing the compactness weight superpixels tend to be more regular in shape and have similar spatial extent. This also comes at a cost, namely that the region mean images do not represent the input image as well as the one for the non-compact version, but are merely a subsampled version.

Segmenting a 'Texture-Only' Image. Our final (extreme) experiment shows how our approach compares to the method of Veksler et al. and SLIC when applied to a synthetic image where all regions have the same mean gray value but differ in variance. Figure 5 (top row) depicts that SLIC fails to extract any useful superpixels, while the method of Veksler et al. has problems especially in regions

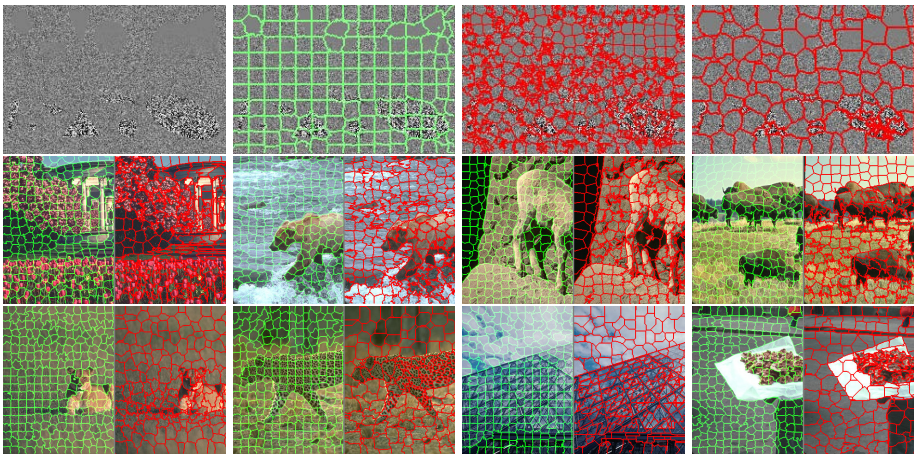

Fig. 5. Top row: Texture (or noise) sensitivity: Test using an image with regions differing only in variance, exploring the case of highly textured regions. Left to right: input image, results from proposed method, `SLIC` and `vekslerConst`. **Middle and bottom row** show typical results for highly textured scenes obtained with the proposed method (green boundaries) compared to `SLIC` (red boundaries, second row), and `vekslerConst` (red boundaries, third row), respectively. Note that all methods were initialised with the same number of superpixels, but `SLIC` and `vekslerConst` may reduce the number of superpixels in the final result. Best viewed in color.

of high variance. While our method also does not give fully satisfactory results in this extreme case, it can be seen that, due to the emphasis on statistically homogeneity instead of smoothness, the proposed method does not spend 'energy' on random region contours and regards textured areas as such. Furthermore, Fig. 5 (row 2 and 3) shows that this is not a contrived experiment, but that this effect is also present in natural images.

7 Conclusions

We have enhanced the *contour-relaxed superpixels* approach and performed an extensive qualitative and quantitative evaluation of this method which is based on a statistical image model and a simple, but efficient optimization scheme. Due to the new compactness term, our approach performs, in terms of standard superpixel benchmarks, comparable, mostly even better than state-of-the-art approaches such as SLIC or the Veksler et al. method. This suggests that the choice of the energy function (derived from a statistical model in our case) might be at least equally important as the optimization method. Computationally, the proposed approach compares similar or favorably against leading state-of-the-art methods. The design parameters of the method allow to tune its behavior in a goal-directed way between high precision of boundaries, strong homogeneity of the region-internal texture, and smoothness/compactness of boundaries.

Acknowledgements. This work was supported by the German Federal Ministry of Education and Research (BMBF) in the project Bernstein Fokus Neurotechnologie – Frankfurt Vision Initiative 01GQ0841, and in parts supported by the ELLIIT programme funded by the Swedish Government.

References

1. Ren, X., Malik, J.: Learning a classification model for segmentation. In: ICCV (2003)
2. Ochs, P., Brox, T.: Object segmentation in video: A hierarchical variational approach for turning point trajectories into dense regions. In: ICCV (2011)
3. Wang, S., Lu, H., Yang, F., Yang, M.: Superpixel tracking. In: ICCV (2011)
4. Gorelick, L., Delong, A., Veksler, O., Boykov, Y.: Recursive MDL via graph cuts: Application to segmentation. In: ICCV (2011)
5. Mori, G., Ren, X., Efros, A.A., Malik, J.: Recovering human body configurations: combining segmentation and recognition. In: CVPR, pp. 326–333 (2004)
6. Comaniciu, D., Meer, P.: Mean shift: A robust approach toward feature space analysis. PAMI 24, 603–619 (2002)
7. Vincent, L., Soille, P.: Watersheds in digital spaces: an efficient algorithm based on immersion simulations. PAMI 13, 583–598 (1991)
8. Levinshtein, A., Stere, A., Kutulakos, K., Fleet, D., Dickinson, S., Siddiqi, K.: Turbopixels: Fast superpixels using geometric flows. PAMI (2009)
9. Veksler, O., Boykov, Y., Mehrani, P.: Superpixels and supervoxels in an energy optimization framework. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part V. LNCS, vol. 6315, pp. 211–224. Springer, Heidelberg (2010)
10. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Suesstrunk, S.: SLIC Superpixels. Technical Report Nr. 149300, EPFL, Lausanne (CH) (2010)
11. Shi, J., Malik, J.: Normalized cuts and image segmentation. TPAMI 22 (2000)
12. Felzenszwalb, P., Huttenlocher, D.: Efficient graph-based image segmentation. IJCV (2004)
13. Moore, A., Prince, S., Warrell, J., Mohammed, U., Jones, G.: Superpixel lattices. In: CVPR (2008)
14. Zhang, Y., Hartley, R., Mashford, J., Burn, S.: Superpixels via pseudo-boolean optimization. In: ICCV (2011)
15. Mester, R., Conrad, C., Guevara, A.: Multichannel segmentation using contour relaxation: Fast super-pixels and temporal propagation. In: Heyden, A., Kahl, F. (eds.) SCIA 2011. LNCS, vol. 6688, pp. 250–261. Springer, Heidelberg (2011)
16. Guevara, A., Conrad, C., Mester, R.: Boosting segmentation results by contour relaxation. In: ICIP (2011)
17. Besag, J.: On the statistical analysis of dirty pictures. Journal of the Royal Statistical Society, Series B 48, 259–302 (1986)
18. Besag, J.: Spatial interaction and the statistical analysis of lattice systems. Journal of the RSS, Series B 36, 192–236 (1974)
19. Martin, D., Fowlkes, C., Tal, D., Malik, J.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: ICCV, vol. 2, pp. 416–423. IEEE (2001)
20. Arbelaez, P., Maire, M., Fowlkes, C., Malik, J.: Contour detection and hierarchical image segmentation. TPAMI (2010)