Int J Comput Vis (2012) 99:53–68
DOI 10.1007/s11263-012-0518-7

A Linear Framework for Region-Based Image Segmentation and Inpainting Involving Curvature Penalization

Thomas Schoenemann · Fredrik Kahl · Simon Masnou · Daniel Cremers

Received: 18 February 2011 / Accepted: 16 February 2012 / Published online: 3 March 2012

Abstract We present the first method to handle curvature regularity in region-based image segmentation and inpainting that is independent of initialization.

To this end we start from a new formulation of length-based optimization schemes, based on surface continuation constraints, and discuss the connections to existing schemes. The formulation is based on a *cell complex* and considers basic regions and boundary elements. The corresponding optimization problem is cast as an integer linear program.

We then show how the method can be extended to include curvature regularity, again cast as an integer linear program. Here, we are considering pairs of boundary elements to reflect curvature. Moreover, a constraint set is derived to ensure that the boundary variables indeed reflect the boundary of the regions described by the region variables.

We show that by solving the linear programming relaxation one gets reasonably close to the global optimum, and that curvature regularity is indeed much better suited in the presence of long and thin objects compared to standard length regularity.

Keywords Curvature regularity · Image segmentation · Inpainting · Linear programming · Cell complexes

T. Schoenemann (✉) · F. Kahl
Center for Mathematical Sciences, Lund University, Lund, Sweden
e-mail: thomas_schoenemann@yahoo.de

S. Masnou
Institut Camille Jordan, Université Lyon 1, CNRS, Villeurbanne, France

D. Cremers
Department of Computer Science, TU München, München, Germany

1 Introduction

Regularization is of central importance for many inverse problems in computer vision including image segmentation and inpainting (Mumford and Shah 1989; Chan and Vese 2001; Masnou and Morel 1998; Bertalmío et al. 2001; Tschumperlé 2006; Bornemann and März 2007; Cao et al. 2012). The introduction of higher-order regularizers in respective energy minimization approaches is known to give rise to substantial computational challenges. Some of the most powerful approaches to image segmentation are based on region integrals with regularity terms defined on the region boundaries (Blake and Zissermann 1987; Mumford and Shah 1989; Nitzberg et al. 1993; Boykov and Jolly 2001; Chan and Vese 2001; Esedoglu and March 2003; Klodt et al. 2008). While many such methods make use of length as a regularity term, only few use curvature regularity. This is in contrast to psychophysical experiments on contour completion (Kanizsa 1971) where curvature was identified as a vital part of human perception. Note that curvature regularity is qualitatively different from length regularity. As shorter boundary curves are preferred in the length case, this causes the well-known shrinking bias. This is not the case for curvature, since the total curvature of any closed, convex curve is equal to 2π.

Length regularization has become an established paradigm as there exist many powerful algorithms for computing optimal solutions for such energy functionals, either using discrete graph-theoretic approaches based on the min-cut/max-flow duality (Greig et al. 1989; Boykov and Jolly 2001) or using continuous PDE-based approaches using convex relaxation and thresholding theorems (Nikolova et al. 2006). Region-based problems for segmentation using curvature regularity have typically been optimized using local optimization methods only (cf. Nitzberg et al. 1993;

Fig. 1 With the proposed method, long and thin structures are much better handled than with length-based approaches

Fig. 2 *Curvature regularity* on the level lines improves inpainting

Esedoglu and March 2003). As a consequence, experimental results highly depend on the choice of initialization. Moreover, these methods do not offer any insights concerning how close the computed solution is to the (unknown) global solution.

In this paper, we propose a relaxed version of region-based segmentation which can be solved optimally. The key idea is to cast the problem of region-segmentation with curvature regularity as an integer linear program (ILP). By solving its LP-relaxation and thresholding the solution we obtain a solution to the original integer problem and are able to evaluate a bound on its quality with respect to the globally optimal solution. In addition, we show that the method readily extends to the problem of inpainting.

Figure 1 demonstrates that the proposed method allows segmenting objects in a way which preserves perceptually important thin and elongated parts. The found solution is within 1.3% of the global optimum. Figure 2 demonstrates the superior performance of curvature regularity over length regularity in a corresponding inpainting experiment.

Existing Work on Curvature Regularity. For contour- or edge-based segmentation methods researchers have successfully developed algorithms to optimally impose curvature regularity using shortest path approaches (Amini et al. 1990) or ratio cycle formulations (Schoenemann and Cremers 2007) on a graph representing the product space of image pixels and tangent angles (Parent and Zucker 1989). In the region-based settings considered, curvature is usually handled by local evolution methods (Chan et al. 2002; Esedoglu and March 2003; Nitzberg et al. 1993; Tschumperlé 2006). Among the methods that pre-date our conference publication (Schoenemann et al. 2009), the only exception we are aware of is the inpainting approach of Masnou and Morel (1998) who can optimize the L_1-norm of the curvature in the absence of regional data terms using dynamic programming.

In this paper we propose an LP-relaxation approach to minimize curvature in region-based settings. In contrast to Masnou and Morel (1998) it allows imposing arbitrary functions of curvature and arbitrary data terms. The algorithmic formulation is based on the concepts of *cell complexes* and *surface continuation constraints* which have been pioneered by Sullivan (1994) and Grady (2010) in the context of 3D-surface completion.

The present paper is based on our preliminary work in Schoenemann et al. (2009), but contains several novelties. Firstly, we show that the constraint system in Schoenemann et al. (2009) needs to be augmented by additional constraints in order to ensure that the boundary of the region-based segmentation is correctly estimated. Secondly, we discuss the connections of our method, when restricted to length

regularity, with standard methods for length regularity and compare experimentally. In addition, the original inpainting method has improved by incorporating a boundary estimation scheme.

There has been a subsequent paper by El-Zehiry and Grady (2010) which optimizes the same model as in our original paper, but applies quadratic pseudo-Boolean optimization (QPBO) with ternary cliques for obtaining the solution. In the case of a square grid, QPBO is able to efficiently compute a solution with curvature regularization, and with a subsequent probing stage often the global optimum is found. However, the discretization artefacts are severe for a cell complex of squares. For better connectivities one has to use fourth order cliques (Strandmark and Kahl 2011), and the resulting optimization problem becomes much harder (QPBO rarely gets close to the global optimum).

A preliminary version of this work was previously presented at a conference (Schoenemann et al. 2009). A similar treatment of boundaries and cooccurrence of edges (albeit at longer range) was more recently also adopted in the context of object detection (Toshev et al. 2010). More recently, a spatially continuous relaxation of curvature regularity on the basis of Menger-Melnikov curvature has been proposed (Goldlücke and Cremers 2011).

The source code associated to this paper is freely available under the name **RegionCurv** at http://www.maths.lth.se/matematiklth/personal/tosch/download.html and at https:github.com/PetterS/regioncurv. The results were produced with release version 0.8, but the latest release (0.905) with the option `-boundary-constraints simple` gives equivalent results. To allow future researchers an easy comparison, we will also provide binaries on our homepage.

2 Length-Based Segmentation Problems

To detail the proposed method for curvature regularity, we first introduce a novel method for length-based segmentation problems. In practice there are more efficient algorithms for this problem (Boykov and Kolmogorov 2003; Nikolova et al. 2006), but in contrast to them the presented one is easily extended to curvature. A comparison to the known techniques is given at the end of this section.

Given an image $I : \Omega \to \mathbb{R}$, the problem is to segment it into two regions, foreground and background. Here "region" means an arbitrary subset of Ω, i.e. there can be several disconnected components and each one can have holes. Hence, each point $\mathbf{x} \in \Omega$ is to be assigned a region $u(\mathbf{x}) \in \{0, 1\}$ where 0 denotes background, 1 foreground.

The desired segmentation is defined as the global optimum of an energy function, consisting of two terms. The first one is called the *data term* and specified by a function $g_0(\mathbf{x})$ for points belonging to the background and a function $g_1(\mathbf{x})$ for the foreground. Both functions will generally depend on the input image I. In addition there is a *regularity* term that penalizes the length of the segmentation boundary by a weighting parameter $\nu \geq 0$. The arising energy minimization problem to be solved is

$$\min_{u:\Omega\to\{0,1\}} \int_\Omega g_0(\mathbf{x})\big[1 - u(\mathbf{x})\big]\,d\mathbf{x} + \int_\Omega g_1(\mathbf{x})\,u(\mathbf{x})\,d\mathbf{x} + \nu|C|, \tag{1}$$

where $C = \partial\{\mathbf{x} \mid \nabla u(\mathbf{x}) = \mathbf{0}\}$ is the set of points where u is discontinuous (i.e. "jumps" from 0 to 1) and $|C|$ denotes its one-dimensional measure. In other words, C is a set of closed lines (which may include parts of the boundary of Ω) and $|C|$ denotes the sum of the length of all lines.

For convenience, we reformulate (1) by splitting the integrand of the first term into a constant term and one depending on u. Defining $g(\mathbf{x}) = g_1(\mathbf{x}) - g_0(\mathbf{x})$ the resulting functional is

$$\min_{u:\Omega\to\{0,1\}} \int_\Omega g(\mathbf{x})u(\mathbf{x})\,d\mathbf{x} + \nu|C| + \text{const.} \tag{2}$$

In the following we will ignore the constant except when evaluating relative gaps between a lower bound and the energy of some segmentation.

2.1 Discretization

In this paper we consider *discretized* segmentation problems where instead of optimizing infinitely many values $\{u(\mathbf{x}) \mid \mathbf{x} \in \Omega\}$ we only consider finitely many "basic regions", henceforth called *cells*, and jointly assign all points in a cell to the same segment. Note that in practice we will always get a discrete input image I, where the cells are given by pixels. Hence, the data term by itself will produce such an assignment even for the continuous problem. This is no longer true when the regularity term is added, but in practice the discretized energy function can be designed to account for this phenomenon (Boykov and Kolmogorov 2003; Nikolova et al. 2006).

We require that our set of cells—denoted $\mathcal{F}$—be a *cell complex* and a partitioning of Ω, i.e. that (1) no two cells overlap and (2) the union of all cells yields Ω. An example is given in Fig. 3(a).

The presented approach makes use of another essential part of a cell complex: *boundary segments*. These are the line segments that form the borders of the cells. Usually a boundary segment has two neighboring cells, except for segments at the border of Ω where there is only one. The set of all boundary segments is denoted $\mathcal{E}$. As will be shown below we need to consider both possible ways of traversing a boundary segment. Hence, for each boundary segment we consider two *oriented boundary segments*, also called *line*

segments in the following. The set of all these segments is denoted $\mathcal{E}^O$ and $\ell(e)$ will denote the length of a line segment e.

In essence, the data term will be defined in terms of the cells, the regularity term in terms of the boundary segments. To approximate the continuous problem sufficiently well, cells should generally not correspond to pixels. Instead, as detailed in Fig. 4 we split the pixels into either 4 or 32 cells, induced by lines with 8 and 16 different directions respectively. In the following we will refer to this as 8- and 16-connectivity.

2.2 An Integer Linear Program

The presented method casts a discretized version of (1) as a so-called *integer linear program*, i.e. minimizing a linear cost function over integral variables and subject to linear constraints. There are two sets of variables: firstly, for each cell $f \in \mathcal{F}$ there is a *region variable* $\mathbf{y}_R^f \in \{0, 1\}$ with 0 indicating that the cell belongs to the background, 1 to the foreground. The second set contains a *boundary variable* $\mathbf{y}_B^e \in \{0, 1\}$ for every oriented boundary segment $e \in \mathcal{E}^O$. Here we want $\mathbf{y}_B^e$ to be 1 only if exactly one of the adjacent cells belongs to the foreground. Now we are already in a position to express the cost function:

$$\mathbf{c}_R^T \mathbf{y}_R + \mathbf{c}_B^T \mathbf{y}_B, \tag{3}$$

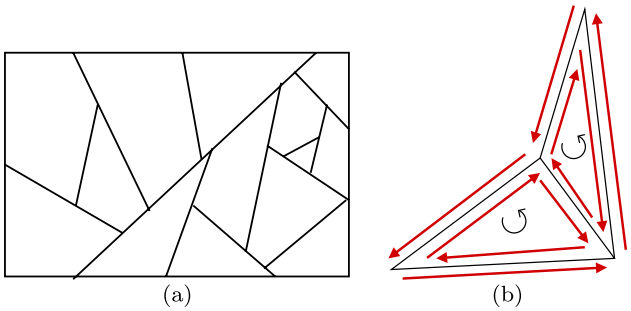


Fig. 3 The basic concepts of our method. (**a**) A cell complex. (**b**) The method considers oriented cells and oriented boundary elements

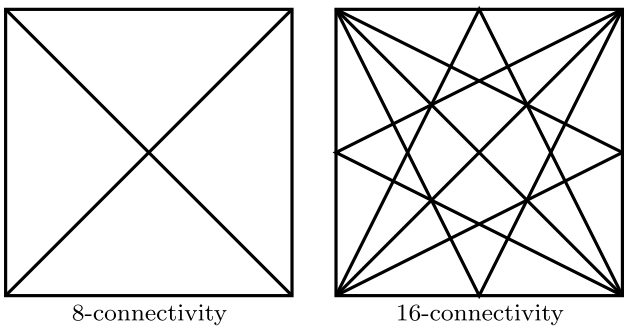


Fig. 4 Splitting a pixel into cells using lines with 8 and 16 different directions

where $\mathbf{c}_R$ contains entries

$$\mathbf{c}_R^f = \int_f g(\mathbf{x})\, d\mathbf{x},$$

and $\mathbf{c}_B$ contains entries $c_B^e = \nu\, \ell(e)$. Since in our case cells are always subsets of a single pixel this weight is simply the function $g(\cdot)$ evaluated at the pixel times the area of the cell.

As ν is positive minimizing (3) by itself would set all boundary variables to 0, so we need constraints. Indeed, up to a few ambiguities (see next section) the region boundary is completely specified by the region variables and the boundary variables serve to render the cost function linear. They are forced to describe the correct boundary by the linear constraint system that we now describe. In words it can be stated as

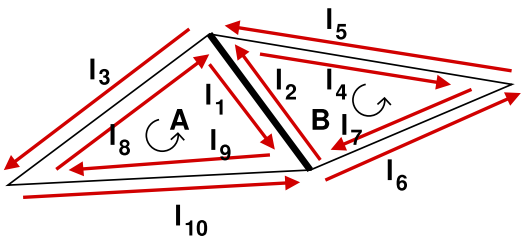
> **Surface Continuation Constraint:** *Whenever a cell is part of the foreground, along each of its boundary segments the foreground must either continue with another foreground cell or with an appropriately oriented boundary segment.*

Formalizing these constraints (one for each boundary segment) involves the concept of *orientations* for both cells and boundary segments. For boundaries we have already introduced this concept, but it is essential for the constraint system that we (arbitrarily) define a "positive" and a "negative" orientation for each boundary segment.

For a cell, an orientation denotes one of the two possibilities for traversing its boundary line—clockwise or counterclockwise. Here it is essential that all cells have the same orientation.

Now, to formalize the surface continuation constraint we define the notion of *positive* and *negative* incidence of cells $f \in \mathcal{F}$ and line segments $l \in \mathcal{E}^O$ to boundary segments $e \in \mathcal{E}$. Ultimately the constraint will then state that the weighted sum of "active" cell incidences must be equal to the weighted sum of "active" boundary incidences, where *active* refers to elements where the associated indicator variable is 1.

While introducing the employed notation we illustrate it on the following example, where we focus on the bold boundary segment, called **e**, and assign **e** an upward orientation.



Incidence of cells and boundary segments is defined for all pairs of cells f and boundary segments e, in terms of an integer value $m_e^f \in \{-1, 0, 1\}$. For a specific e, most cells f will not border on e, which is denoted by $m_e^f = 0$. Since we are operating on a two dimensional cell complex, at most two cells can have a non-zero value for a fixed e. Whether this is a 1 or a -1 is defined by the orientation of the cell: if, when traversing the boundary segments of a cell in the associated orientation, one traverses e in its assigned (positive) orientation, we set $m_e^f = 1$. Otherwise, i.e. when traversing e in the opposite (negative) orientation, we set $m_e^f = -1$. In our example above, we have $m_{\mathbf{e}}^A = 1$ and $m_{\mathbf{e}}^B = -1$. If the cell complex contained more cells, then for any such f we would have $m_{\mathbf{e}}^f = 0$.

The incidence of a line l and a boundary segment e is denoted by the integer $m_e^l \in \{-1, 0, 1\}$ and again defined for all pairs of line segments and edges. It is 0 for almost all such pairs, solely the two oriented line segments belonging to an edge e take a non-zero value m_e^l. Here the one that agrees with e in orientation gets a coefficient of $m_e^l = 1$, the oppositely oriented edge a coefficient of $m_e^l = -1$. In our example, we have $m_{\mathbf{e}}^{l_1} = -1$ and $m_{\mathbf{e}}^{l_2} = 1$. For all other line segments l_k where $k \geq 3$ we have $m_{\mathbf{e}}^{l_k} = 0$.

The constraint system can now be stated as

$$\sum_{f \in \mathcal{F}} m_e^f \, y_R^f = \sum_{l \in \mathcal{E}^O} m_e^l \, y_B^l \quad \forall e \in \mathcal{E}. \tag{4}$$

In our example, the constraint for the bold edge reads

$$y_R^A - y_R^B = y_B^{l_2} - y_B^{l_1}. \tag{5}$$

In the case where $y_R^A = 1$ and $y_R^B = 0$, i.e. cell A is foreground and cell B is background, $y_B^{l_2}$ will be forced to 1, whereas $y_B^{l_1}$ will be set to 0. If instead B is foreground and A background, this will force $y_B^{l_1}$ to be 1 and $y_B^{l_2}$ to be 0. If both A and B belong to the same component, the constraint leaves some freedom for the boundary variables: they can now both be 0 or both be 1. The latter is undesirable, but will not happen as long as the length weight is strictly positive. However, when we integrate curvature below we will need extra constraints to prevent this case.

Finally, we summarize the integer linear program to be solved as:

$$\begin{aligned} \min_{\mathbf{y}} \quad & \mathbf{c}_R^T \mathbf{y}_R + \mathbf{c}_B^T \mathbf{y}_B \\ \text{s.t.} \quad & \sum_{f \in \mathcal{F}} m_e^f \, y_R^f = \sum_{l \in \mathcal{E}^O} m_e^l \, y_B^l \quad \forall e \in \mathcal{E} \\ & y_R^f \in \{0, 1\} \quad \forall f \in \mathcal{F} \\ & y_B^l \in \{0, 1\} \quad \forall l \in \mathcal{E}^O. \end{aligned} \tag{6}$$

As we will show now this problem can be efficiently solved by computing a graph min-cut.

2.3 Relation to Graph Cuts

Discrete approaches to image segmentation are very well studied in computer vision and the vast majority uses pixels as their cells. However, they do usually not express (length) regularity in terms of the boundary segments of these cells—this would imply a four-connectivity.

Instead, these approaches are based on *graphs* where the cells correspond to nodes and the length term is represented in terms of *edges* that connect pairs of nodes (Boykov and Kolmogorov 2003). We denote the set of nodes $\mathcal{P}$, where each $p \in \mathcal{P}$ represents the center point of a cell $f \in \mathcal{F}$. Analogous to the above integer program, each center p of a cell is associated a binary variable $y_p \in \{0, 1\}$ indicating foreground and background. The smoothness term is modeled by a set of edges $\mathcal{N}$ (also called neighborhood) and for length regularity can be expressed as:

$$|C| \approx \frac{1}{k(\mathcal{N})} \sum_{(p,q) \in \mathcal{N}} \frac{1}{\|p - q\|} \big(1 - \delta(y_p, y_q)\big),$$

where $k(\mathcal{N})$ is a normalization constant[1] that ensures that the weights remain comparable when the size of the neighborhood is enlarged.

Minimizing this energy can be written as a *minimum cut* problem, which again can be written as

$$\min_{\mathbf{y}_P \in \{0,1\}^{|\mathcal{P}|}} \quad \mathbf{c}_R^T \mathbf{y}_P + \sum_{(p,q) \in \mathcal{N}} w_{p,q} \, |y_p - y_q|.$$

It is well-known, e.g. (Dantzig and Thapa 1997), that such absolutes can be rewritten as linear programs, i.e. this problem can be equivalently written as

$$\begin{aligned} \min_{\mathbf{y}_P, \mathbf{a}_\pm} \quad & \mathbf{c}_R^T \mathbf{y}_P + \sum_{(p,q) \in \mathcal{N}} w_{p,q} \, (a_{p,q}^+ + a_{p,q}^-) \\ \text{s.t.} \quad & y_p - y_q = a_{p,q}^+ - a_{p,q}^- \quad \forall (p,q) \in \mathcal{N} \\ & \mathbf{y}_P \in \{0, 1\}^{|\mathcal{P}|}, \quad a_{p,q}^+, a_{p,q}^- \geq 0 \quad \forall (p,q) \in \mathcal{N}. \end{aligned}$$

If we assume that the neighborhood links exactly all pairs of neighboring cells (i.e. those that share a boundary segment), these are exactly our surface continuation constraints (5). Since graph cuts can be optimized globally efficiently it follows that (6) is polynomial-time solvable.

In summary, what we have proposed so far is a restriction of graph cuts to graphs that are planar when source and sink are removed. This restriction is crucial for (our solution of) the problem we really want to solve: to integrate curvature regularity into the framework.

[1] This makes sense only if $\mathcal{N}$ is spatially invariant (except for pixels near the image border), which we assume here. In this case one can set $k(\mathcal{N}) = \sum_{q:(p,q) \in \mathcal{N}} 1/\|p - q\|$, where p is an arbitrary pixel in the image interior.

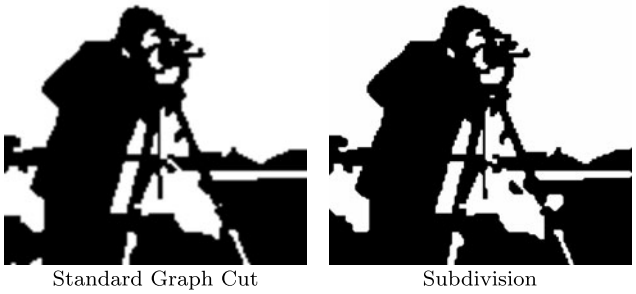


Fig. 5 Comparison of standard graph cuts and the novel subdivision scheme, both run with the same length weight and an 8-connectivity. (Input image shown in Fig. 11.) The major difference is in the lower right quadrant, near the right leg of the tripod

In contrast to standard applications of graph cuts our method relies on subdiving pixels. In the case of a very strong data term a boundary pixel will probably not be split into a foreground and a background part. Standard graph cuts might then be the better way to reflect the length regularity term. However, in practice boundary pixels usually have weak data terms due to partial overlap. Figure 5 shows the resulting segmentations of both schemes, where the input image can be found in Fig. 11. Here it can be seen that the novel scheme gives access to a higher resolution and produces slightly different segmentations. These are actually of lower energy than for the standard scheme. Hence, if anything we have *gained* something with the novel formulation.

2.4 Relation to Sullivan's Method

Sullivan (1992, 1994) gave a method for finding a surface of minimal (weighted) area in 3D-space[2] subject to the constraint that the surface span a given (set of) boundary lines. His method also relies on a cell complex and can be written as a polynomial-time solvable minimum cost flow problem. Further, the method allows integrating volume terms.

In essence, what we have done so far is a restriction of this method to 2D and where the set of prescribed boundary elements is empty (these objects would be one-dimensional). However, there is one major difference: in Sullivan's method all variables are unrestricted,[3] whereas we have the constraints $y_R^f \in \{0,1\}$ and $y_B^l \in \{0,1\}$. And of course our main goal is to integrate curvature regularity.

[2]In fact, he considered general N-D spaces, but this is not important here.

[3]Cf. (Sullivan 1994, page 21, top). Sullivan's volume terms are non-negative.

3 Handling Curvature Regularity

We now show how the described integer linear program can be generalized to curvature regularity, i.e. to the model

$$\min_{u:\Omega\to\{0,1\}} \int_\Omega g(\mathbf{x})\, u(\mathbf{x})\, d\mathbf{x} + \nu\,|C| + \lambda \int_C \left|\kappa_C(\mathbf{x})\right|^p d\mathcal{H}^1(\mathbf{x}). \tag{7}$$

Here $\lambda > 0$ is a curvature weight, $\kappa_C(\mathbf{x})$ stands for the curvature of the line C at a given point $\mathbf{x}$ of the line and $p > 0$ is an arbitrary exponent (usually $p = 2$ is a good choice). The notation $d\mathcal{H}^1(\mathbf{x})$ signifies that the integral is over a set of lines and that it is independent of the parameterization of these lines.

We emphasize that our method allows more general regularity terms, namely arbitrary positive functions depending on position, direction and absolute curvature. In particular this allows spatially weighted regularity terms.

Clearly, the combination of length and curvature introduces another free parameter (λ in addition to ν), which gives more room to adjust to a specific input image. We emphasize that for this work we use ν merely as a stabilizer to speed up the computations. That is, we choose it small enough to not have a significant impact on the results.

3.1 Discretizing the Problem

Again, our solution is based on a cell complex and the data term is handled in the exact same way as above. That is, we again have region indicator variables y_f for all $f \in \mathcal{F}$. We were able to express the length regularity in terms of (single) boundary segments. For curvature, this is not possible: all boundary segments are straight lines, hence have curvature 0 everywhere. The only points where non-zero curvature can occur are the meeting points of two boundary segments. Hence it is common to consider pairs of line segments (Parent and Zucker 1989; Amini et al. 1990) to express curvature regularity. So far, however, this was not compatible with region terms.

For every pair l_1, l_2 of adjacent line segments with compatible orientations we now have an indicator variable $\mathbf{y}_B^{l_1,l_2} \in \{0,1\}$. Here we follow the convention that the line segment that is traversed earlier is also listed first in the pair. Getting back to our example on page 56, we have the pairs (l_1, l_6) and (l_1, l_9) *starting* with l_1, whereas (l_8, l_1) and (l_5, l_1) *end* with l_1. Note that e.g. (l_1, l_7) is *not* a valid pair as the orientations do not match.

We get a cost function of the form

$$\mathbf{c}_R^T \mathbf{y}_R + \mathbf{c}_B^T \mathbf{y}_B, \tag{8}$$

where $\mathbf{y}_B$ now contains all the pairwise variables. We proceed to describe the entries of the corresponding cost vector $\mathbf{c}_B^T$.

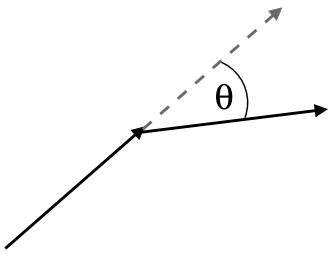


Fig. 6 The angle θ is the basis for computing the curvature of the pair of black lines

3.2 Computing the Weights

Computing curvature from two adjacent line segments is based on considering the direction change, measured by the angle θ in Fig. 6.

There are basically two ways to compute the term $|\kappa|^p$ from this angle: firstly, we can just take the power p of the angle (which should be measured in arc length). The second method is based on the work of Bruckstein et al. (2001), and makes also use of the lengths $\ell(l_1)$ and $\ell(l_2)$ of the two lines:

$$\min\{\ell(l_1),\ell(l_2)\}\left(\frac{\theta}{\min\{\ell(l_1),\ell(l_2)\}}\right)^p.$$

The original idea of Bruckstein et al. was to take the longest straight lines pre- and succeeding a direction change. In our context we can only consider elementary boundary segments, so we do not have the same convergence properties. In practice we found that both weights work fine and Bruckstein et al's weights significantly reduce the running time of the employed linear programming solver.

Denoting either of the two arising weights as w_{l_1,l_2}, we get one of two components for the cost entry c_{l_1,l_2}. Together with length regularity this entry is

$$c_{l_1,l_2} = w_{l_1,l_2} + \tfrac{1}{2}\big(\ell(l_1)+\ell(l_2)\big).$$

There are however some special cases involving the image border, a point that we neglected in Schoenemann et al. (2009): firstly, at the four corners of a rectangular domain Ω we will want to set the curvature weight to 0. Note that we still penalize the angle in which a region boundary meets the domain boundary $\partial\Omega$.

A second point relates to the length weight: whenever one (or two) of the line segments in a pair is part of the domain border, its length should be set to 0. Otherwise there would be a bias towards associating the cells at the image border to the background.

3.3 An Adequate Constraint System

As before for length regularity, the linear cost function (8) needs to be minimized subject to suitable constraints to closely reflect the discretized model function. In Schoenemann et al. (2009) we presented two sets of constraints, with the aim to ensure that the boundary variables indeed describe a boundary of the region variables. In this work we show that two more sets of constraints are needed, where the second one becomes necessary only when several cells meet in a single point. In particular, we will show that in this case there are several valid boundaries and that our method searches for the one with least cost.

Firstly, we adapt the surface continuation constraints to the new kind of boundary variables. To this end, we define the incidence $m_e^{l_1,l_2}$ of a line segment pair and a boundary segment $e \in \mathcal{E}$. This value is defined as 0 unless l_1 is an orientation of e (note that l_2 is irrelevant for this value). Otherwise the value is the same as the previously defined incidence of l_1 and e. Hence, the surface continuation constraints read

$$\sum_{f\in\mathcal{F}} m_e^f\, y_R^f = \sum_{l_1,l_2\in\mathcal{E}^O} m_e^{l_1,l_2}\, y_B^{l_1,l_2} \quad \forall e\in\mathcal{E}. \tag{9}$$

For our example cell complex on page 56, the surface constraint for the bold edge now reads

$$y_R^A - y_R^B = y_B^{l_2,l_3} + y_B^{l_2,l_4} - y_B^{l_1,l_6} - y_B^{l_1,l_9}. \tag{10}$$

These constraints alone leave a lot of freedom. In particular, we could choose line pairs without direction changes everywhere. Hence, these constraints alone merely account for length regularity. What is really wanted is that for every active $y_B^{l_1,l_2}$ both l_1 and l_2 belong to the region boundary induced by the region variables. This is ensured by two sets of constraints, where the first is called *boundary continuation*. In words it can be stated as

> **Boundary Continuation Constraint:** *If a pair of line segments l_1, l_2 is active, there must be a succeeding pair l_2, l_3 that is also active. Likewise there must be a preceding active pair l_0, l_1.*

These constraints ensure that the active line pairs actually define closed paths. They are identical to the constraints arising for the computation of shortest paths in a graph such as (Amini et al. 1990) and are stated as

$$\sum_{l_0} y_B^{l_0,l_1} = \sum_{l_2} y_B^{l_1,l_2} \quad \forall l_1 \in \mathcal{E}^O.$$

Now we have paths, but we cannot guarantee that all parts of these paths are actually region boundaries. Indeed, we invite the reader to check that the configuration in Fig. 7(a) satisfies all constraints introduced so far. Moreover, for small length weights and squared curvature the cost of this configuration will be lower than those of the desired configuration shown in part (b). To exclude cases such as (a) from the optimization, we add a new set of constraints:

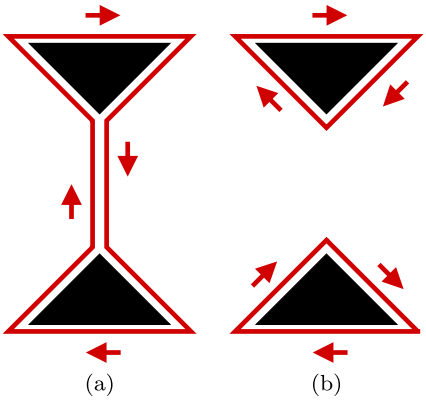


Fig. 7 Without the boundary consistency constraints the configuration in (**a**) is valid and—for squared curvature and without length penalty—cheaper than the desired one in (**b**). With boundary consistency (**b**) remains feasible, but (**a**) is excluded, as desired

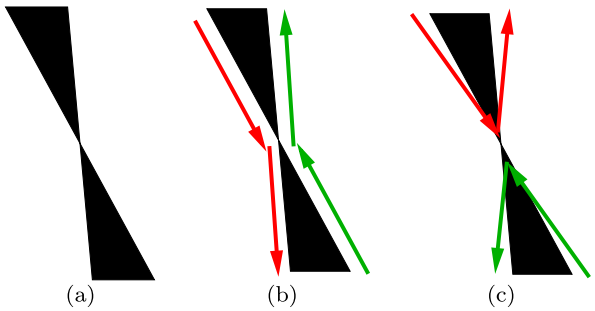


Fig. 8 Determining boundaries is ambiguous: (**a**) a segmentation where two cells meet in a point. (**b**) A low-cost boundary configuration, where different pairs are indicated by different colors. (**c**) An equally valid, but more expensive boundary

Boundary Consistency Constraint: *For every boundary segment, only one of the two possible orientations can be active.*

To formalize this, we denote by $e^{\rightarrow}$ and $e^{\leftarrow}$ the positive and negative orientation of a boundary segment e. Then the constraint can be written as

$$\sum_{l_1} y_B^{l_1,e^{\leftarrow}} + \sum_{l_2} y_B^{e^{\rightarrow},l_2} \leq 1 \quad \forall e \in \mathcal{E}.$$

Similar sets of constraints can be derived, e.g. $\sum_{l_1} y_B^{e^{\leftarrow},l_1} + \sum_{l_2} y_B^{l_2,e^{\rightarrow}} \leq 1$, but experimentally we found them to be redundant. Moreover, if e is part of the domain border $\partial\Omega$ then one of the orientations will never occur in a desired configuration. We simply disregard the corresponding variables.

Recently, Strandmark and Kahl (2011) proposed an alternative constraint system to ensure consistency. This turns out to give a tighter relaxation and is available in RegionCurv version 0.905.

With these constraints, there is still one issue left to take care of, and it affects cases where several cells meet in a point. In this case, as shown in Fig. 8, there are several valid boundary configurations. As long as only two cells meet, the given constraint system ensures that indeed the configuration with lower cost is selected. However, as soon as there are three or more cells meeting in a point, this constraint system will allow configurations *with crossings*, as exemplified in Fig. 9. Should we really avoid such configurations? There is actually no obvious answer if one has in mind that, in the theory of continuous plane curves, the set enclosed by a positively oriented closed curve can be defined as the points of positive *index* (also called *winding number*) (Rudin 1987). In the example of Fig. 9, the index of each point in a black triangle with respect to the outer curve defined by the arrows is one, so it makes sense to consider the curve as a region boundary.[4] For the sake of comparison and completeness, we provide in Sect. 6, Fig. 13 a couple of experiments done either with or without crossing prevention. In the former case, we add a new set of constraints:

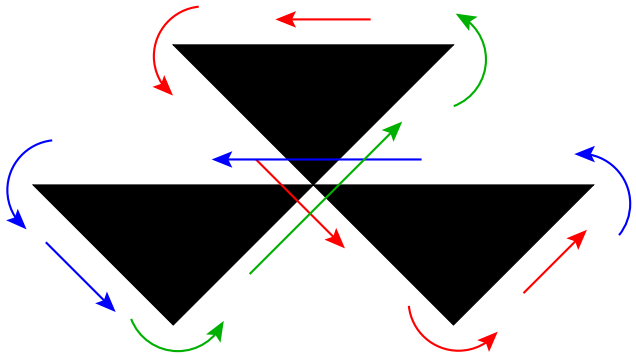


Fig. 9 If three or more cells meet in a point, self-intersecting boundaries can define valid segmentations. If these are undesired, a fourth constraint set is needed. The different colors denote different phases when traversing the line. There is only one line

Crossing Prevention Constraint: *If two pairs of line segments cross, only one of them may be active.*

Denoting $\mathcal{C}$ the set of crossing line pairs, this constraint is easily formalized as

$$y_B^{l_1,l_2} + y_B^{l_3,l_4} \leq 1 \quad \forall (l_1,l_2,l_3,l_4) \in \mathcal{C}.$$

In practice these constraints are ignored in a first phase. Afterwards, the (usually very few) violated constraints are added in passes and the system is re-solved, until there are no more violated constraints. In our experiments we never needed more than 9 passes. However, solving the arising programs can be quite time consuming, even when starting from the previous configuration.

[4] The index of a point on a discrete grid with respect to a discrete curve can also be computed, see for instance the winding number algorithm at www.softsurfer.com/Archive/algorithm_0103/algorithm_0103.htm.

In summary, region-based segmentation with curvature regularity is now expressed as the integer linear program

$$\begin{aligned}
\min_{\mathbf{y}_R,\mathbf{y}_B} \quad & \mathbf{c}_R^T \mathbf{y}_R + \mathbf{c}_B^T \mathbf{y}_B \\
\text{s.t.} \quad & \sum_{f\in\mathcal{F}} m_e^f\, y_R^f = \sum_{l_1,l_2\in\mathcal{E}^O} m_e^{l_1,l_2}\, y_B^{l_1,l_2} \quad \forall e\in\mathcal{E} \\
& \sum_{l_0} y_B^{l_0,l_1} = \sum_{l_2} y_B^{l_1,l_2} \quad \forall l_1\in\mathcal{E}^O \\
& \sum_{l_1} y_B^{l_1,e^{\leftarrow}} + \sum_{l_2} y_B^{e^{\rightarrow},l_2} \le 1 \quad \forall e\in\mathcal{E} \\
& y_R^f \in\{0,1\} \quad \forall f\in\mathcal{F}, \quad y_B^{l_1,l_2}\in\{0,1\} \quad \forall l_1,l_2\in\mathcal{E}^O,
\end{aligned}$$

and the optional constraints

$$y_B^{l_1,l_2} + y_B^{l_3,l_4} \le 1 \quad \forall (l_1,l_2,l_3,l_4)\in\mathcal{C}.$$

A first evaluation of this new scheme was given in Fig. 1 which shows that curvature regularity is better suited to ensure connected regions in the presence of long and thin objects. Note that the found solutions for curvature are generally not globally optimal. Details on the optimization scheme are given in Sect. 5 and more experiments will be given in Sect. 6. First, however, we discuss how to handle the problem of image inpainting.

4 Inpainting

In image inpainting we are given an image $I:\Omega\to\mathbb{R}$ together with a damaged region $\Omega_d\subset\Omega$. This damaged region can have arbitrarily many connected components and each of these can enclose holes. The task is to fill the damaged region with values that fit nicely with the values of I outside the damaged region.

To this end, the above integer linear program is generalized to give a *structured* inpainting approach, where we consider the continuous model (Masnou and Morel 1998; Masnou 2002; Chan et al. 2002)

$$\begin{aligned}
&\min_{u:\Omega\to\{0,1\}} \int_{I_l}^{I_u} \int_{C_{u,t}} \left|\kappa_{C_{u,t}}(\mathbf{x})\right|^p d\mathcal{H}^1(\mathbf{x})\, dt \qquad (11)\\
&\text{s.t.} \quad u(\mathbf{x}) = I(\mathbf{x}) \quad \forall \mathbf{x}\in\Omega\setminus\Omega_d,
\end{aligned}$$

where I_l and I_u are the minimal and maximal intensities of I along the border of the damaged region, $C_{u,t}=\{\mathbf{x}\mid u(\mathbf{x})=t\}$ is the set of level lines for level t of $u(\cdot)$ and again $p>0$ is an exponent for curvature.

For the case of absolute curvature ($p=1$) and that Ω_d consists of *singly*-connected components (i.e. the components do not enclose holes), an efficient global optimization scheme was given in Masnou and Morel (1998), Masnou (2002). We are interested in the more general problem of arbitrary domains and exponents $p>0$. In particular, $p=2$ is usually a better model.

4.1 Discretization

As before for image segmentation, our strategy is to discretize the model (11) by introducing cells and pairs of boundary segments. However, the cells are no longer limited to two labels: the intensity u_f of cell $f\in\mathcal{F}$ can be anywhere between I_l and I_u. We follow the strategy in Masnou and Morel (1998), Masnou (2002) and consider only integral values inside this range. The result is a fully discrete labeling problem.

Naturally, one also needs to change the right-hand-side values of the inequality constraints. Moreover, for inpainting we always include the crossing prevention constraints since by definition level lines cannot cross.

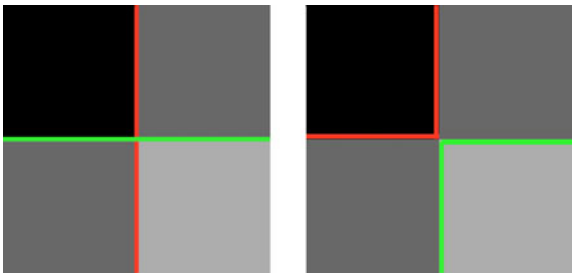
To make sure that the boundary variables truly reflect level lines, it would be advisable to associate each cell multiple variables, reflecting level sets. Then, there would be a *binary* variable y_f^k for every integral value $I_l\le k\le I_u$ where a value of 1 reflects that the intensity u_f of the cell is at least equal to k (i.e. $u_f\ge k$). For $k'>k$ this would naturally entail the constraints $y_f^{k'}\ge y_f^k$. Moreover, there would be binary variables y_{l_1,l_2}^k that would be forced to be consistent with the level variables exactly as for binary segmentation.

This strategy is however not practicable for the domain sizes we want to address: the problem is way too large scale. Recently Schoenemann et al. (2011) managed to reduce the problem to a practicable size by grouping intensities into bins, a compromise between totally neglecting the problem and correctly handling it (available in RegionCurv version 0.905).

For the present work, we settle for one integral variable $y_R^f\in\{I_l,\ldots,I_u\}$ for every cell $f\in\mathcal{F}$, directly reflecting the intensity u_f of the face. In addition, there are boundary variables reflecting the intensity differences between neighboring cells. Inside the damaged domain they can be restricted to values $y_B^{l_1,l_2}\in[0,I_u-I_l]$. For the fixed part, we only consider cells that border on the damaged region. At their other borders the respective boundary variables can take values in $\{0,\ldots,I_u\}$. In practice it is advisable to first subtract the constant I_l from the entire image (note that each connected component of Ω_d can be processed independently).

As shown[5] in Fig. 10 this strategy does generally not give level lines, but it is a reasonable approximation. The arising

[5] Many thanks to Yubin Kuang for providing these images.



(a) (b)

Fig. 10 Best viewed in color. Estimated boundary variables for the given intensity profiles. Colors denote different pairs (not all pairs are shown) (**a**) With the proposed approximation we do not always get level lines. Here the boundary variables may jump between different intensity levels. (**b**) With level variables one would get the correct solution, but the computational demands are too high in practice

integer linear program is stated as

$$\begin{aligned}
\min_{\mathbf{y}_R,\mathbf{y}_B} \quad & \mathbf{c}_B^T \mathbf{y}_B && (12)\\
\text{s.t.} \quad & \sum_{f\in\mathcal{F}} m_e^f\, y_R^f = \sum_{l_1,l_2\in\mathcal{E}^O} m_e^{l_1,l_2}\, y_B^{l_1,l_2} && \forall e\in\mathcal{E}\\
& \sum_{l_0} y_B^{l_0,l_1} = \sum_{l_2} y_B^{l_1,l_2} && \forall l_1\in\mathcal{E}^O\\
& \sum_{l_1} y_B^{l_1,e^{\leftarrow}} + \sum_{l_2} y_B^{e^{\rightarrow},l_2} \le I_u && \forall e\in\mathcal{E}\\
& y_B^{l_1,l_2} + y_B^{l_3,l_4} \le I_u && \forall (l_1,l_2,l_3,l_4)\in\mathcal{C}\\
& y_R^f = I_f - I_l && \forall f\subset\Omega\setminus\Omega_d\\
& y_R^f \in \{0,\ldots,I_u - I_l\} && \forall f\subset\Omega_d\\
& y_B^{l_1,l_2} \in \{0,\ldots,I_u - I_l\} && \forall l_1,l_2\in\mathcal{E}^O.
\end{aligned}$$

Note that we have one such program for every connected component of Ω_d.

4.2 Estimating Incoming Level Lines

A correct estimation of the direction of the level lines touching from outside the inpainting domain Ω is important, since the method aims at prolonging them with no additional curvature, if possible. We follow the simple and efficient method proposed by Bornemann and März (2007) after the work of Weickert (1998, 2003) on the robust determination of coherence directions in an image: at a point $x\in\Omega\setminus\Omega_d$, the coherence direction is the normalized eigenvector associated to the minimal eigenvalue of the structure tensor (Bornemann and März 2007):

$$J(x) = \frac{(K_\rho * (\mathbb{1}\Omega\setminus\Omega_d \nabla I_\sigma \otimes \nabla I_\sigma))(x)}{(K_\rho * \mathbb{1}\Omega\setminus\Omega_d)(x)} \qquad (13)$$

where $*$ denotes convolution, $\mathbb{1}\Omega\setminus\Omega_d$ is the characteristic function of $\Omega\setminus\Omega_d$ and I_σ is defined as

$$I_\sigma = \frac{K_\sigma * (\mathbb{1}\Omega\setminus\Omega_d I)}{K_\sigma * \mathbb{1}\Omega\setminus\Omega_d}, \qquad (14)$$

and K_ρ, K_σ are Gaussian smoothing kernels with standard deviations ρ and σ. Experimentally, setting $\sigma = 1.5$, $\rho = 4$ yields a reliable estimation of the incoming level lines directions along $\partial\Omega_d$.

5 Optimization Strategies

In general, solving integer linear programs is an NP-hard problem (Schrijver 1986). In some cases, where there are linear *in*equality constraints $\mathbf{Ax}\le\mathbf{b}$ and the matrix $\mathbf{A}$ is *totally unimodular* one can find the global optimum by solving the linear programming relaxation (Schrijver 1986), i.e. the problem one obtains when dropping all integrality constraints on the variables. For instance, a constraint $y_i\in\{0,1\}$ will be relaxed to $y_i\in[0,1]$. The arising problem can be solved in (weakly) polynomial time using interior point methods (Ye 1997).

Several of the discussed systems are in fact polynomial-time solvable, in particular the length-based problems and the boundary continuation constraints by themselves. The integer linear program for curvature regularity is however not in this class. Still, experimentally we found that solving the linear programming relaxation often gives nearly integral solutions. Moreover, the relaxation value provides a (usually quite tight) lower bound on the original problem. We proceed to discuss this strategy in detail.

5.1 Solving the Linear Programming Relaxation

There are two popular ways of solving general linear programs. Firstly, there is the dual simplex method (Dantzig and Thapa 1997), based on refactorizing the constraint system. It is usually the most memory saving method and in practice superior to the primal simplex method. For some variants of the simplex method exponential worst-case run-times have been proved. On practical problems the method often works very well, and there are competitive and freely available implementations, e.g. the solver Clp.[6] We found it quite useful for an 8-connectivity, but for a 16-connectivity—and very low length weights—we got acceptable running times only for some of the images we tried. In other cases the solver was terminated after several days without having solved the problem. In the end we chose the length weights high enough to get acceptable running times.

[6] http://www.coin-or.org/projects/Clp.xml.

The results we get indicate that these settings actually provide a very good model.

On the other hand, there are interior point methods which usually perform Newton-iterations on a primal-dual formulation of the problem. This entails frequent matrix inversions, solved via the sparse Cholesky decomposition. We are not aware of a freely available solver that performs well on large scale problems such as ours. Hence, we tested several commercial packages and found Gurobi and FICO Xpress to be well-suited solvers for our problem. Generally the running times of commercial interior point solvers are quite predictable. The downside is the memory consumption: we found that a little more than twice the memory consumption of a simplex solver is needed for our problem, where we tested with an 8- and a 16-connectivity.

For segmentation, the combination of licensing issues and the high memory demands made us use the simplex method. For inpainting, where the memory demands are lower, the interior point methods proved superior.

5.2 Obtaining and Evaluating Integral Solutions

Solving the linear programming relaxation provides a fractional solution as well as a lower bound on the original integral problem. Since the fractional solutions are often close to integral, we derive an integral solution by simply thresholding the region variables. This already defines a segmentation, but we would also like to know its energy, so we have to infer the associated boundary variables.

We already discussed the complications in case two or more cells meet in a point (see Sect. 3.3). Our method will then select the cheapest allowed boundary configuration according to the selected constraint set. In contrast, methods based on low-order factors (El-Zehiry and Grady 2010; Strandmark and Kahl 2011) will take the sum of all configurations here.

Due to these difficulties we so far did not implement a specialized routine to compute the energy of a segmentation, although this should be possible. Instead, we simply re-run the linear programming solver, this time with all region variables fixed according to the segmentation. Note that this strategy was also pursued in Schoenemann et al. (2009), so the gaps reported there are w.r.t. the integer program, not the model itself.

In case crossing prevention was selected we again add violated constraints in passes. In addition, we fix all "impossible" boundary variables to 0, where impossible refers to pairs of line segments where along one of the line segments the segmentation stays constant. In the vast majority of cases this produced an integral solution and hence the optimal boundary configuration. In a few cases we got up to 12 fractional variables. We presently assume that the computed cost is close enough to the actual cost.

6 Experiments

In this section we evaluate the proposed scheme for both image segmentation and inpainting, where in all cases we consider the intensity range $[0, 255]$. For image segmentation we also evaluate how close to the global optimum we got. Here, we report relative gaps between the computed lower bound and the computed energy of the derived integral solution. A word of warning is appropriate here: the relative gaps depend on the order of magnitude of the optimal energy, i.e. adding a constant to the energy will reduce the relative gap. For many problems in computer vision a natural energy scale is given. However, this is not the case for image segmentation, at least not for data-terms based on squared differences, where the mean-values are kept fix rather than optimized.

In a second point, we report the percentage of variables that were assigned fractional values to illustrate that the major part is assigned integral labels.

The experiments were run on a 3.0 GHz Core2 Duo machine equipped with 8 GB of memory. For segmentation we used the dual simplex method in the solver Clp, for inpainting we used the interior point method of Gurobi.

6.1 Image Segmentation

For binary image segmentation, we show experiments for a totally unsupervised problem and an interactive one where seed nodes are given.

Unsupervised Image Segmentation For unsupervised image segmentation we use data functions g_0, g_1 as in the piecewise constant functional of Mumford and Shah (1989), i.e. where the intensity of an image point is compared to the mean value of a region. This results in the model:

$$\int_\Omega \big(I(\mathbf{x}) - \mu_0\big)^2 \big[1 - u(\mathbf{x})\big]\, d\mathbf{x} + \int_\Omega \big(I(\mathbf{x}) - \mu_1\big)^2 u(\mathbf{x})\, d\mathbf{x} + \nu|C| + \lambda \int_C \big|\kappa_C(\mathbf{x})\big|^2\, d\mathcal{H}^1(\mathbf{x}). \tag{15}$$

We set the mean values μ_0, μ_1 to the minimal and maximal intensity in the given image, respectively.

Figure 11 shows results of our method on images of size 128×128 and 160×107, respectively, using an 8-connectivity. Here we provide results for different curvature weights and it can be seen that even for very high weights long and thin structures are preserved. At the same time, the relative gap increases with the curvature weight. Surprisingly, the percentage of fractional variables decreases.

These results took roughly 4 hours computing time, where up to 9 passes were needed. Though we re-used the existing solution, the first passes often took as long as solving the initial program.

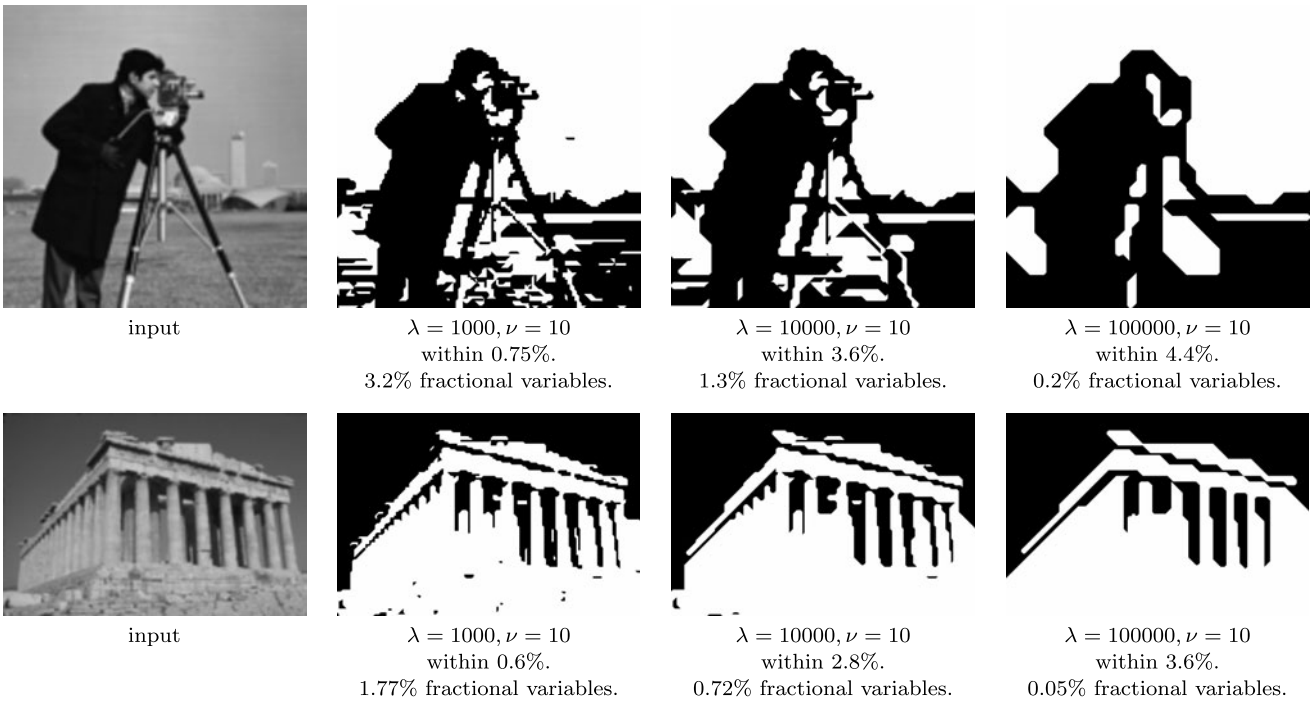


Fig. 11 Evaluation of the proposed method for unsupervised image segmentation and an 8-connectivity. Shown are segmentations for different curvature weights and how close they are to the lower bound (and hence the global optimum)

Interactive Image Segmentation Next we turn to the problem of interactive image segmentation for color images, where in addition to an image $I : \Omega \to \mathbb{R}^3$ we are given a set of foreground and background seed nodes as specified by a user. Since our focus is on evaluating the novel method, we did not refine the seed nodes. Hence, for each image the seed nodes were specified only once.

From the given seed nodes, we estimate normalized histograms of color values, resulting (after smoothing) in distributions $p_F(\cdot)$ and $p_B(\cdot)$ that are then used in the model:

$$-\int_\Omega \log\big(p_F\big(I(\mathbf{x})\big)\big)\big[1-u(\mathbf{x})\big]\,d\mathbf{x}, \tag{16}$$

$$-\int_\Omega \log\big(p_B\big(I(\mathbf{x})\big)\big)\,u(\mathbf{x})\,d\mathbf{x} + \nu|C| + \lambda\int_C \big|\kappa_C(\mathbf{x})\big|^2\,d\mathcal{H}^1(\mathbf{x}). \tag{17}$$

This function is minimized over all $u : \Omega \to [0, 1]$ that are consistent with the seed nodes.

For the experiments, we use a 16-connectivity and fix the ratio of the curvature weight λ over the length weight ν to 20.0. The presented results were then obtained within half a day and using between 6 and 8 GB of memory.

Figure 12 compares our results with a simple thresholding scheme and the length based method of Sect. 2. These results clearly show the tendency of length-based methods to suppress long and thin objects. With curvature regularity this is remedied.

Effects of the Constraint Sets Above we indicated that it is debatable whether self-intersecting region boundaries should be allowed or not. In Fig. 13 we explore both of these possibilities as well as the constraint system without boundary consistency we used in Schoenemann et al. (2009). For these images there were significant differences, for others, like the giraffe image, none at all. For the boat the results clearly improve when forbidding self-intersections, for the other image they grow slightly worse.

Evidently, adding boundary consistency has only a minor impact on the thresholded solution. In fact, the relaxation values increased only marginally. However, the relative gaps as well as the percentage of fractional variables increases with the number of enforced constraints, so clearly the formulations are not equivalent. When using the constraints proposed in Strandmark and Kahl (2011) instead of our boundary consistency constraints, we expect that these figures will reduce and the integral solutions found for the different constraint systems will differ more significantly.

For unsupervised image segmentation Fig. 14 shows that adding boundary consistency does have an effect: the number of sharp angles is reduced when adding the constraints. Also, the relaxation values differ significantly, here. Still, adding the constraints from Strandmark and Kahl (2011) should help here, too.

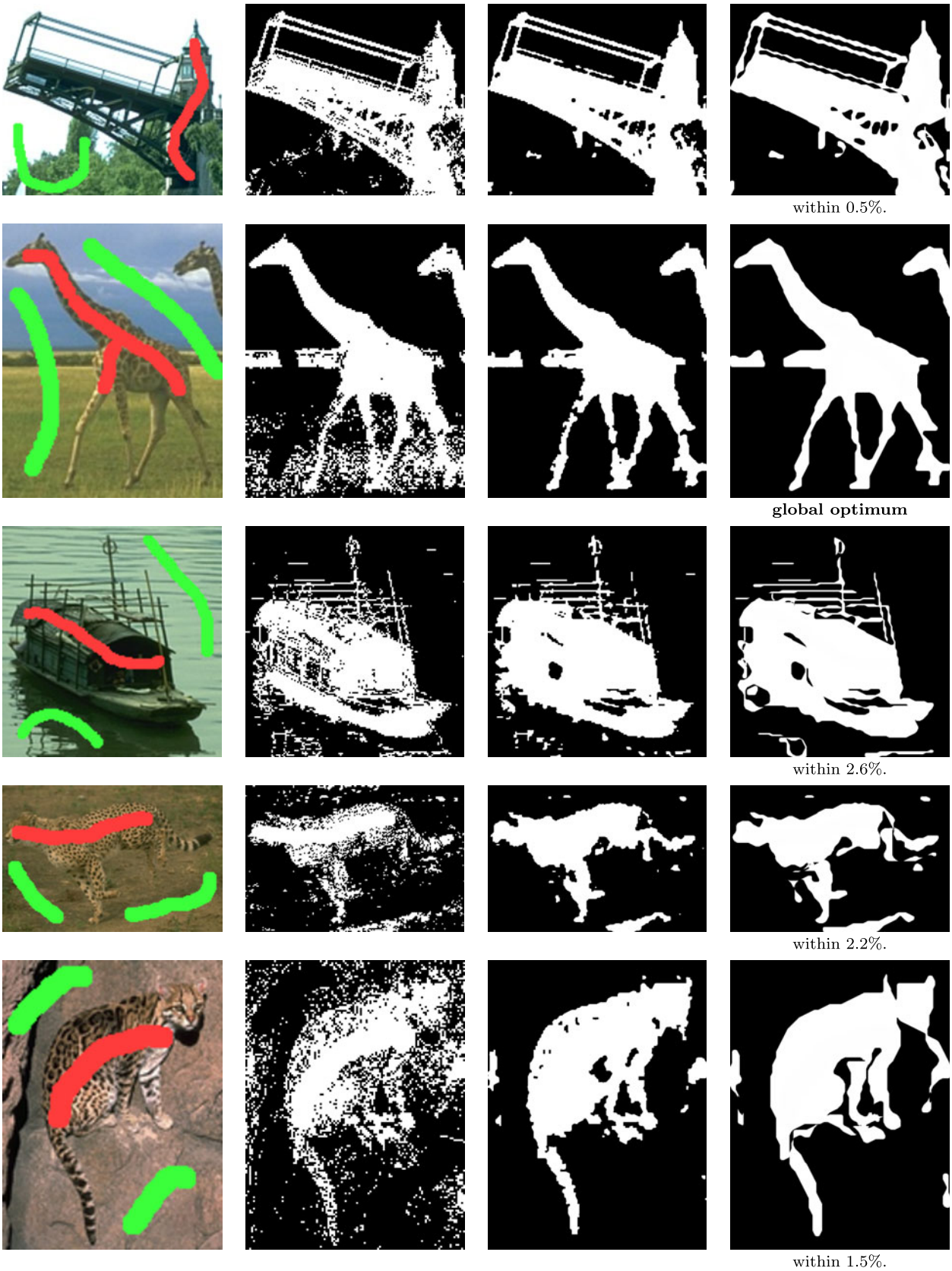


Fig. 12 Comparison of length-based and curvature-based methods. *Left column*: input images with seed nodes super-imposed. *Middle left*: thresholding scheme. *Middle right*: length-based segmentation (proposed method). In all cases $\nu = 1$. *Right*: length- and curvature-based segmentation (proposed method) and how close the results are to the global optimum. In all cases we set $\lambda = 4$ and $\nu = 0.2$. Images taken from the Berkeley database http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/

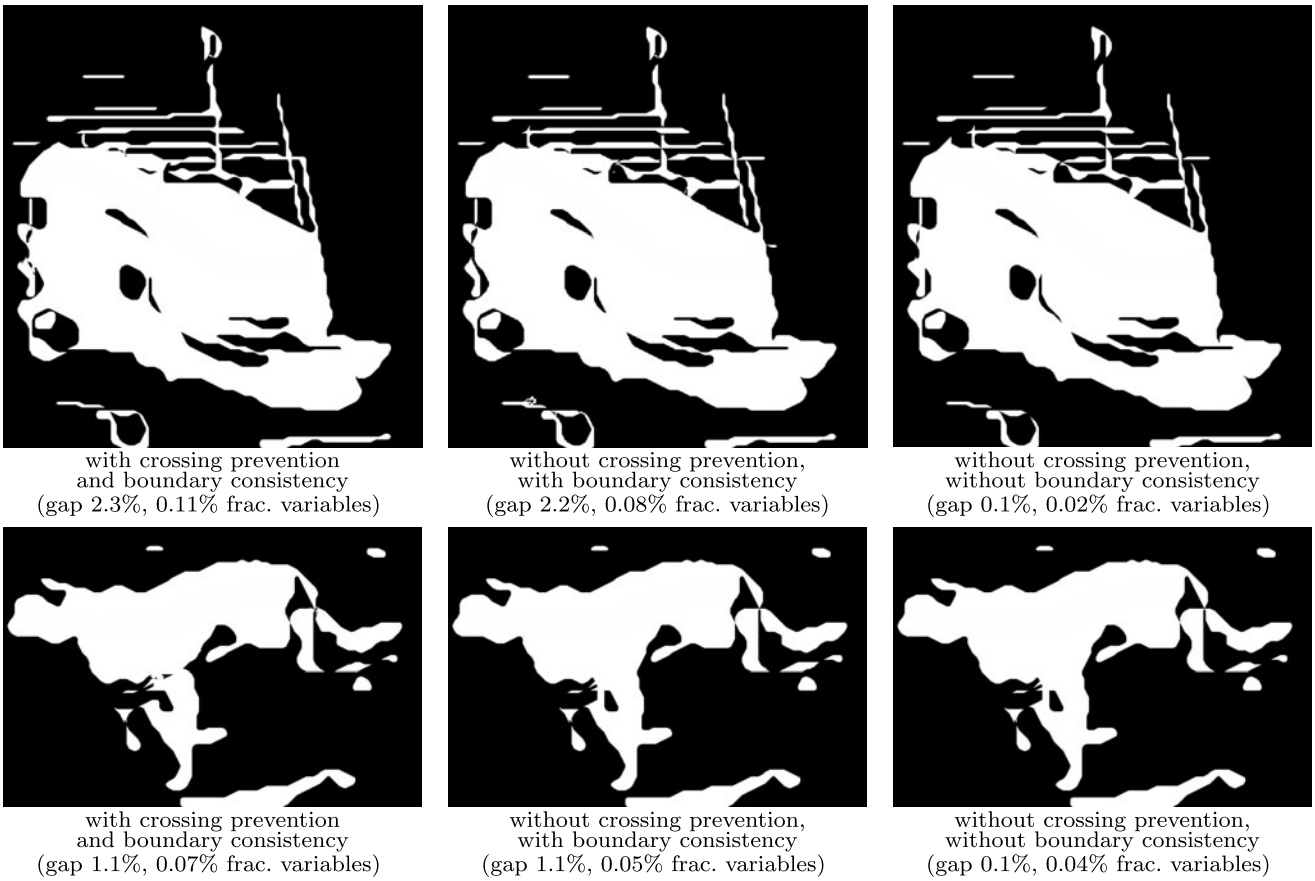


Fig. 13 Without crossing prevention there is a bias towards triple points. For the *top row* significantly different segmentations are obtained with and without the constraints. For the *bottom row* there are only minor changes

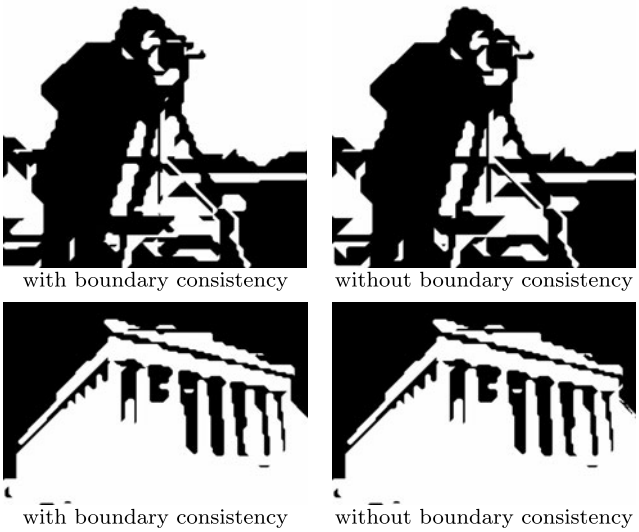


Fig. 14 Unsupervised image segmentation with and without boundary consistency. The number of sharp angles decreases when adding consistency. We used a length weight of $\nu = 10000$, an 8-connectivity and crossing prevention

6.2 Inpainting

We now turn to the problem of inpainting, where we use interior point solvers. Figures 15 and 16 show that our method is well-suited for structured inpainting, and that length regularity generally does not work well.

In this work we have improved upon our work (Schoenemann et al. 2009) by previously estimating the direction of incoming level lines and giving a tighter constraint system. Figure 17 shows that these changes really improve the results.

7 Conclusion

We have presented new theory and methods for length- and curvature-based regularization, both for image segmentation and inpainting. For curvature (in a region-based context) we are the first to propose a global approach in the sense that it is independent of initialization.

The results clearly demonstrate that curvature regularity outperforms length-regularity in the presence of long and

Fig. 15 For inpainting curvature is much better suited than length regularity

Fig. 16 Our method applied to structured inpainting. Images taken from http://www.robots.ox.ac.uk/~vgg/data/data-various.html

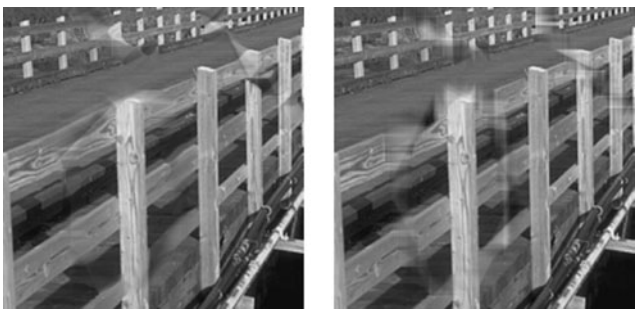


Fig. 17 Comparison of the proposed inpainting method (*left column*) and the one we proposed in Schoenemann et al. (2009) (*right column*). The domain is as in Fig. 15

thin objects. Experimentally we showed that for segmentation our strategy of solving a linear programming relaxation is usually within 5% of the global optimum. In some cases it even finds the global optimum.

Acknowledgements We thank Petter Strandmark and Yubin Kuang for helpful discussions, as well as anonymous reviewers for their constructive comments.

Thomas Schoenemann and Fredrik Kahl were funded by the Swedish Foundation for Strategic Research (SSF) through the programmes Future Research Leaders and Wearable Visual Information Systems, and by the European Research Council (GlobalVision grant no. 209480). Simon Masnou would like to acknowledge funding by the French ANR *Freedom* project. Daniel Cremers' work was financed through the ERC starting grant "ConvexVision".

References

Amini, A. A., Weymouth, T. E., & Jain, R. C. (1990). Using dynamic programming for solving variational problems in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *12*(9), 855–867.

Bertalmío, M., Sapiro, G., Caselles, V., & Ballester, C. (2001). Image inpainting. In *ACM SIGGRAPH*, July 2001.

Blake, A., & Zissermann, A. (1987). *Visual reconstruction*. Cambridge: MIT press.

Bornemann, F., & März, T. (2007). Fast image inpainting based on coherence transport. *Journal of Mathematical Imaging and Vision*, *28*(3), 259–278.

Boykov, Y., & Jolly, M. P. (2001). Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images. In *IEEE international conference on computer vision (ICCV)*, Vancouver, Canada, July 2001.

Boykov, Y., & Kolmogorov, V. N. (2003). Computing geodesics and minimal surfaces via graph cuts. In *IEEE international conference on computer vision (ICCV)*, Nice, France, October 2003.

Bruckstein, A. M., Netravali, A. N., & Richardson, T. J. (2001). Epi-convergence of discrete elastica. *Applicable Analysis*, *79*, 137–171. Bob Caroll Special Issue.

Cao, F., Gousseau, Y., Masnou, S., & Pérez, P. (2012). Geometrically guided expemplar-based inpainting. *SIAM Journal on Imaging Sciences*, submitted.

Chan, T. F., Kang, S. H., & Shen, J. (2002). Euler's elastica and curvature based inpaintings. *SIAM Journal on Applied Mathematics*, *2*, 564–592.

Chan, T. F., & Vese, L. (2001). Active contours without edges. *IEEE Transactions on Image Processing*, *10*(2), 266–277.

Dantzig, G. B., & Thapa, M. N. (1997). *Linear programming 1: introduction*. *Springer series in operations research*. Berlin: Springer.

El-Zehiry, N. Y., & Grady, L. (2010). Fast global optimization of curvature. In *IEEE computer society conference on computer vision and pattern recognition (CVPR)*, San Francisco, California, June 2010.

Esedoglu, S., & March, R. (2003). Segmentation with depth but without detecting junctions. *Journal of Mathematical Imaging and Vision*, *18*, 7–15.

Goldlücke, B., & Cremers, D. (2011). Introducing total curvature for image processing. In *IEEE international conference on computer vision (ICCV)*, Barcelona, Spain, November 2011.

Grady, L. (2010). Minimal surfaces extend shortest path segmentation methods to 3d. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *32*(2), 321–334.

Greig, D. M., Porteous, B. T., & Seheult, A. H. (1989). Exact maximum *a posteriori* estimation for binary images. *Journal of the Royal Statistical Society Series B Methodological*, *51*(2), 271–279.

Kanizsa, G. (1971). Contours without gradients or cognitive contours. *Giornale Italiano Di Psicologia*, *1*, 93–112.

Klodt, M., Schoenemann, T., Kolev, K., Schikora, M., & Cremers, D. (2008). An experimental comparison of discrete and continuous shape optimization methods. In *European conference on computer vision (ECCV)*, Marseille, France, October 2008.

Masnou, S. (2002). Disocclusion: A variational approach using level lines. *IEEE Transactions on Image Processing*, *11*, 68–76.

Masnou, S., & Morel, J. M. (1998). Level-lines based disocclusion. In *International conference on image processing (ICIP)*, Chicago, Michigan, 1998.

Mumford, D., & Shah, J. (1989). Optimal approximations by piecewise smooth functions and associated variational problems. *Communications on Pure and Applied Mathematics*, *42*, 577–685.

Nikolova, M., Esedoglu, S., & Chan, T. (2006). Algorithms for finding global minimizers of image segmentation and denoising models. *SIAM Journal on Applied Mathematics*, *66*(5), 1632–1648.

Nitzberg, M., Mumford, D., & Shiota, T. (1993). Filtering, segmentation and depth. In *LNCS*, vol. 662. Berlin: Springer.

Parent, P., & Zucker, S. W. (1989). Trace inference, curvature consistency, and curve detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *11*(8), 823–839.

Rudin, W. (1987). *Real and complex analysis*. New York: McGraw-Hill.

Schoenemann, T., & Cremers, D. (2007). Introducing curvature into globally optimal image segmentation: minimum ratio cycles on product graphs. In *IEEE international conference on computer vision (ICCV)*, Rio de Janeiro, Brazil, October 2007.

Schoenemann, T., Kahl, F., & Cremers, D. (2009). Curvature regularity for region-based image segmentation and inpainting: A linear programming relaxation. In *IEEE international conference on computer vision (ICCV)*, Kyoto, Japan, September 2009.

Schoenemann, T., Kuang, Y., & Kahl, F. (2011). Curvature regularity for multi-label problems—standard and customized linear programming. In *International workshop on energy minimization methods in computer vision and pattern recognition*, St. Petersburg, Russia, July 2011.

Schrijver, A. (1986). *Theory of linear and integer programming*. *Wiley-Interscience series in discrete mathematics and optimization*. New York: Wiley.

Strandmark, P., & Kahl, F. (2011). Curvature regularization for curves and surfaces in a global optimization framework. In *International workshop on energy minimization methods in computer vision and pattern recognition*, St. Petersburg, Russia, July 2011.

Sullivan, J. M. (1992). A crystalline approximation theorem for hypersurfaces. PhD thesis, Princeton University, Princeton, New Jersey.

Sullivan, J. M. (1994). Computing hypersurfaces which minimize surface energy plus bulk energy. In *Motion by mean curvature and related topics* (pp. 186–197).

Toshev, A., Taskar, B., & Daniilidis, K. (2010). Object detection via boundary structure segmentation. In *IEEE computer society conference on computer vision and pattern recognition (CVPR)*, San Francisco, California, June 2010.

Tschumperlé, D. (2006). Fast anisotropic smoothing of multi-valued images using curvature-preserving PDE's. *International Journal of Computer Vision*, *68*(1), 65–82.

Weickert, J. (1998). *Anisotropic diffusion in image processing*. Stuttgart: Teubner.

Weickert, J. (2003). Coherence-enhancing shock filters. In *Pattern recognition (proc. DAGM)*, Magdeburg, Germany, September 2003.

Ye, Y. (1997). *Interior point algorithms: theory and analysis*. *Wiley-Interscience series in discrete mathematics*. New York: Wiley.