Uninitialized, Globally Optimal, Graph-Based Rectilinear Shape Segmentation — The Opposing Metrics Method[*]

Ali Kemal Sinop
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213
asinop@cmu.edu

Leo Grady
Department of Imaging and Visualization
Siemens Corporate Research
Princeton, NJ 08540
leo.grady@siemens.com

Abstract

We present a new approach for the incorporation of shape information into a segmentation algorithm. Unlike previous approaches to the problem, our method requires no initialization, is non-iterative and finds a steady-state (i.e., global optimum) solution. In the present work, we are specifically focused on the segmentation of rectilinear shapes. The key idea is to use the fact that certain shape classes optimize the ratio of specific metrics, which can be expressed as graph Laplacian matrices applied to indicator vectors. We show that a relaxation of the binary formulation of this problem allows a global solution via generalized eigenvectors. The approach is tested on both synthetic examples and natural images.

1. Introduction

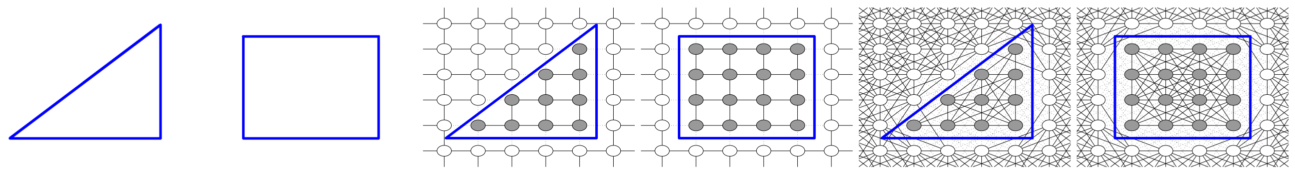
In recent years there has been a trend toward formulation of image segmentation algorithms as variational problems in which the minimal energy solution corresponds to the desired segmentation. A primary advantage of a variational formulation, in the context of shape segmentation, is the availability of an energy that provides a measure of closeness between the segmentation and the desired shape. This measure is extremely important since occlusion, noise or image deformation often distorts the object in the image such that an exact match with the desired shape is impossible. When an exact match is impossible, a variational formulation allows the algorithm to find a segmentation of the "best" match, with respect to the energy functional.

Variational approaches in the literature can be roughly divided into two groups: 1) Functionals that require an initial solution and produce iterative improvement toward a local minimum (usually via gradient descent), 2) Functionals for which a global minimum is efficiently achievable. The first type of approach is necessary when the formulation of the relevant functional is nonconvex, nonlinear or nondifferentiable. However, when possible, functionals of the second type are generally preferred since problems with initialization, local minima and the particulars of optimization are avoided. Although the first category encompasses many algorithms, the second class seems to have been almost universally phrased in the language of discrete optimization on a graph (e.g., [15, 11, 2, 10]).

The addition of high-level shape information has long been recognized to benefit more general low-level segmentation methods for particular segmentation tasks. Previous efforts to incorporate shape information into variational algorithms have all been drawn from the first class of approaches in which iterative improvements are made toward a local minimum. Examples of this type of algorithm include active shape models [4], OBJ CUT [13], level sets [6] etc. In contrast, the algorithm proposed here belongs to the second type of variational algorithm in the sense that it will be possible to achieve a global minimum (of the binary-relaxed problem) without initialization. As with the majority of other algorithms belonging to the second type, we adopt a combinatorial (graph) formulation. We note that while the work of Freedman [7] also develops a variational shape-based algorithm with a global optimum (although the energy function is sampled at various scales), in contrast to the present approach, Freedman's method requires user interaction (in the form of seeds) and is not translation invariant. Additionally, the work of Werner [16] defines rectilinear shapes on a graph as the optimum of an energy with a global minimum (of the relaxed problem). Unfortunately, this approach has no tolerance to even slight rotation and is extremely computationally intensive.

The particular preferred shape that we target for segmentation in this paper is the rectilinear shape. Rectilinear shapes occur in many practical segmentation problems

[*]Published in Proc. of ICCV 2007, Rio de Janeiro, Brazil, Oct., 2007



(a) 4-unit triangle. ℓ_1 perimeter: 16, ℓ_2 perimeter: 13.6569 (b) 4-unit square. ℓ_1 perimeter: 16, ℓ_2 perimeter: 16 (c) Cut-measured perimeter on L_1 graph: 16 (d) Cut-measured perimeter on L_1 graph: 16 (e) Cut-measured perimeter on L_2 graph: 11.8317 (f) Cut-measured perimeter on L_2 graph: 13.9192

Figure 1. A rectilinear shape is known to minimize the ratio of the perimeter measured with an ℓ_1 metric and the perimeter measured with an ℓ_2 metric [18] as shown in the top row. It is possible to measure the perimeter of a object, expressed as a cut, with respect to various metrics with arbitrary accuracy by adjusting the graph topology and weighting [3]. The Opposing Metrics strategy for segmenting rectilinear shapes is to look for the segmentation that minimizes the cut with respect to one graph, L_1, while maximizing the cut with respect to a second graph, L_2.

associated with applications involving man-made structures such as buildings, roadsigns or factory environments. We emphasize that we are looking for the shape that best fits the rectilinear energy function, rather than strictly axis aligned *rectangles* with a known scale or aspect ratio. If a scale, rotation and aspect ratio were known, then a Hough-type algorithm could be used for the segmentation. Even if one were willing to endure the computational expense of a search over all scales, rotations and aspect ratios with a Hough-type approach, such a method would not return the "most rectilinear" object in the image, in the event that there was no object that was strictly rectilinear.

This new variational algorithm has the feature that it is uninitialized, produces a global minimum (of the binary-relaxed problem), is efficiently optimized and provides a measure of the rectilinearity of the resulting segmentation. As with other algorithms of the second class of variational algorithms presented above, our algorithm is formulated on a graph and takes advantage of discrete optimization techniques. Due to the combinatorial formulation, it is possible to precisely measure the effect of resolution (scale) on the effectiveness of the shape information.

Our algorithm makes use of previous work on rectilinear shape measures [18] to establish rectilinearity as the ratio of the object perimeter measured by an ℓ_1 and an ℓ_2 metric. Since the rectilinearity measure is derived from contrasting the measure of the object perimeter by two different metrics, we term the algorithm the **opposing metrics** (OM) algorithm. The object segmentation is represented by an indicator vector on the set of nodes (pixels) with a perimeter measured by the *cut* with respect to these metrics, as represented by different graph topologies (using the results in [3]). The binary formulation is then relaxed to allow a global optimization to solve for the indicator vector. The resulting optimization may be done by a generalized eigenvector computation. Although eigenvector computations are well-known in the context of normalized cuts segmentation [15], we stress that they develop their algorithm from a different functional and produce a different segmentation from our algorithm that does not prefer rectilinear shapes. On a final note, we would like to mention that this formulation also extends to higher dimension in the sense that 3D cubes could also be used as preferences for a 3D segmentation algorithm. Due to space limitations, this 3D material will not be further explored in the present work.

2. Method

In this section, we begin by defining a rectilinearity measure, provide a representation of the segmentation that permits optimization, formulate the segmentation algorithm with the data term and discuss details of optimization.

2.1. Rectilinearity measure

In order to incorporate shape information into a variational segmentation algorithm, it is first necessary to provide a method for measuring how well a given segmentation result fits the shape model. Measures of rectilinearity have been previously proposed, due to the importance of rectilinear segmentation in real-world applications. One such measure was introduced by Zunic and Rosin [18] who showed that the only class of shapes which minimize the following ratio

$$Q(P) = \frac{\mathrm{Per}_1(P)}{\mathrm{Per}_2(P)}, \tag{1}$$

are rectilinear shapes (with respect to the specified X and Y axis). Here $\mathrm{Per}_1(P)$ denotes the ℓ_1 perimeter of shape P according to the specified X and Y axis. Similarly, $\mathrm{Per}_2(P)$ denotes the perimeter of shape P with the ℓ_2 metric. Formally,

$$\begin{aligned} \mathrm{Per}_1(P) &= \int |u'(t)| + |v'(t)| dt, & (2)\\ \mathrm{Per}_2(P) &= \int \sqrt{u'^2(t) + v'^2(t)} dt, & (3) \end{aligned}$$

for some parametrization $u(t), v(t)$ of the shape boundary P. We note that (1) does not provide a rotationally invariant measure of rectilinearity (i.e., it is assumed that the rectangular object roughly aligns with the image axes). This point will be revisited again from a practical standpoint in Section 3.2. Our goal now is to formulate our segmentation in such a way that the shape minimizes (1) while also respecting the image data.

2.2. Representation

The formulation of our algorithm will be on a graph. A **graph** consists of a pair $G = (V, E)$ with **vertices (nodes)** $v \in V$ and **edges** $e \in E \subseteq V \times V$, with $N = |V|$ and $M = |E|$. An edge, e, spanning two vertices, v_i and v_j, is denoted by e_{ij}. A **weighted graph** assigns a value to each edge called a **weight**. The weight of an edge, e_{ij}, is denoted by $w(e_{ij})$ or w_{ij} and is assumed here to be nonnegative. The **degree** of a vertex is $d_i = \sum w(e_{ij})$ for all edges e_{ij} incident on v_i. The following will also assume that our graph is connected and undirected (i.e., $w_{ij} = w_{ji}$). An image may be associated with a graph by identifying each pixel with a node and defining an edge set to represent the local neighborhood relationship of the pixels. The edge set (neighborhood) structure will be used to represent the opposing metric spaces.

As with other several other graph-based segmentation algorithms [15, 11], we represent our desired segmentation as an indicator vector that may then be solved for. Define an indicator vector representing a segmentation as

$$x_i = \begin{cases} 1 & \text{if } v_i \text{ is in the object,} \\ 0 & \text{if } v_i \text{ is in the background.} \end{cases} \tag{4}$$

and the graph Laplacian matrix, L, as

$$L_{ij} = \begin{cases} d_i & \text{if } i = j, \\ -w_{ij} & \text{if } v_i \text{ and } v_j \text{ are adjacent nodes,} \\ 0 & \text{otherwise,} \end{cases} \tag{5}$$

where L_{ij} is indexed by vertices v_i and v_j.

Given a segmentation (represented by indicator vector x), the weighted cost of the cut (perimeter), C, may be written as

$$C = x^T L x = \sum_{ij} w_{ij}(x_i - x_j)^2. \tag{6}$$

It was shown by Kolmogorov and Boykov that we may use (6) to give the perimeter of the object with respect to various metrics by designing the appropriate weighting/topology [3]. In the following, we assume that the pixel (node) spacing is at one unit in each direction. To encode ℓ_1 in the graph weighting/topology, we employ a four-connected grid with weighting

$$w_{ij} = \begin{cases} 1 & \text{If } v_i \text{ and } v_j \text{ are adjacent in the grid,} \\ 0 & \text{Otherwise.} \end{cases} \tag{7}$$

Figure 1 illustrates that this design produces the ℓ_1 measure of the object perimeter when measured using (6). We will denote the Laplacian matrix for this topology/weighting as L_1.

Similarly, Boykov and Kolmogorov [3] showed that the length of the boundary using any Riemannian metric can be approximated with an appropriately weighted graph. Hence, if we let L_2 denote the Laplacian matrix of the ℓ_2 metric, $\lim_{\delta \to 0} x^T L_2 x = \mathrm{Per}_2(\Omega x)$ for grid spacing δ and object represented by x. In the remainder of this work, we define the pixels to have unit spacing, i.e., $\delta = 1$. If we let $E_k \subset E$ indicate the edge family incident on node v_k, where all edges are sorted according to their angle ϕ_i, the weight of each edge is computed from [3] as

$$w_i = \frac{\Delta \phi_k}{2|e_i|}, \tag{8}$$

for $e_i \in E_k$. Here, $\Delta \phi_k$ stands for the angle between two edges in E_k, and $|e_i|$ is the (Euclidean) length of this edge. Each node is connected to all of its neighbors within a radius R. From [3] we know that increasing R increases the accuracy of the approximation of the Euclidean measurement of perimeter via cuts. In Section 3.2 we show that a sufficient approximation of the Euclidean metric for our purposes is obtained by employing $R = 3$–4.

Therefore, in terms of an indicator vector on a graph, we may rewrite (1) as

$$\frac{\mathrm{Per}_1(\Omega x)}{\mathrm{Per}_2(\Omega x)} \approx \frac{x^T L_1 x}{x^T L_2 x}. \tag{9}$$

An example of the graphs representing L_1 and L_2 are given in Figure 1 along with sample ratios for a triangle and a rectangle. As desired, the rectangle has a smaller ratio.

Given the above definitions of the Laplacian matrix representing ℓ_1 and ℓ_2 perimeters, we may now define the combinatorial formulation of (10) for rectilinear shapes. Specifically, the segmentation represented by x that optimizes

$$\arg\min_x \frac{x^T L_1 x}{x^T L_2 x}, \tag{10}$$

subject to $\left[x \in \{-1, 1\}^N\right]$ will have a rectilinear shape.

2.3. Image segmentation with rectilinear shape preference

In the previous section, we showed that it is possible to provide a representation of the segmentation (given as an indicator vector) that allows the measure of the rectilinearity of that segmentation. Of course, rectilinearity alone is insufficient to find a rectilinear shape in a particular image — image data must also be incorporated. Our design combines the shape and data information together to guide the algorithm to look for a segmentation that fits data while favoring a rectilinear shape.

To incorporate image information into this segmentation, we adopt an image gradient based approach similar to [5]. Each edge weight of the L_2 graph is multiplied with the following term

$$\tilde{w_{ij}} = \exp^{\frac{\beta}{\eta} \max \text{Gradient}(e_{ij})}. \tag{11}$$

Here $\max \text{Gradient}(e_{ij})$ is intended to indicate the maximum gradient magnitude along the line spanned by edge e_{ij} (see [5] for details). The value of $\beta > 0$ is a free parameter and η represents the normalization term such that $\eta = \max_{e_{ij}} \text{Gradient}(e_{ij})$.

Since we are maximizing with respect to L_2, this modulation of the edge weights forces the segmented object to favor boundaries that align with the image intensity gradients. If the desired application involved the segmentation of textured or colored rectilinear shapes, the gradient of these values would be used in evaluating (11).

2.3.1 Handling the image borders

One concern with the above formulation is that an object segmentation sharing its boundary with the image border has no penalty, i.e., there is no cost associated with labeling a node on the border as foreground or background. This situation encourages the segmented object boundary to coincide with the image borders. Since objects sharing its boundary with the image border should not be favored over interior objects, we add a term representing the image border to (10) in the following way:

$$\arg\min_x \frac{x^T(L_1 + \gamma I_b)x}{x^T L_2 x}, \tag{12}$$

where I_b is a diagonal indicator matrix for nodes on the borders.

2.4. Optimization

In the previous sections, we have developed the formulation in (10) as a method by which a rectilinear, data-respecting object may be segmented from an image. We turn now to the question of how to solve for the segmentation x. Although both the numerator and denominator of (10) are convex functions, optimization of the ratio is non-trivial. In this section, we first discuss a purely combinatorial approach to the optimization before settling on a relaxation of the binary formulation in order to find a minimum.

2.4.1 Rank-1 Relaxation

Following [12], a natural approach to the optimization of (12) might be via Semi-Definite Programming. Although faster methods have been developed for SDP via approximation algorithms [8], such an approach appears to be too slow for practical implementation of the OM algorithm. Our implementation of the approximation method of [8] required roughly two hours to complete the computation for a 40×40 image. Therefore, practical realities insist that we pursue another technique for minimizing (12).

2.4.2 Binary relaxation

A common approach taken to the optimization of a binary formulation of ratios has been to *relax* the binary constraint in order to provide a method for global optimization [15, 11]. Although it is difficult to analytically relate the solution of the relaxed problem to the solution of the binary problem, empirical evidence shows that generally a good solution is found [15, 11].

In order to solve (12), we relax the integer constraint and allow the x_i's to take continuous values over the real line. Noting that a relaxed (12) is the Rayleigh quotient, the optimal value of (12) is achieved by the generalized eigenvalue and eigenvector pair λ, x of equation $L_1 x = \lambda L_2 x$ corresponding to the smallest nonzero generalized eigenvalue.

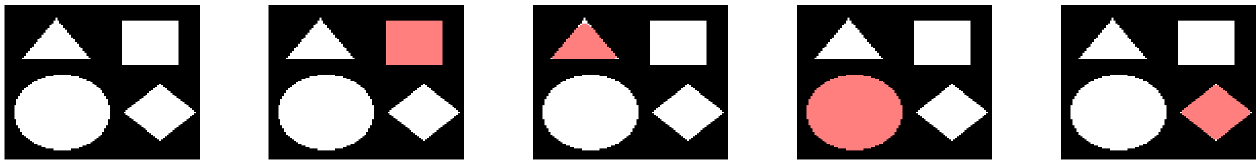
Eigenvector computations of the graph Laplacian for segmentation are well known to the vision community following the introduction of the popular Normalized Cuts algorithm [15]. We wish to stress that our OM method differs significantly from the Normalized Cuts approach since Normalized Cuts demands computation of eigenvectors of the (normalized) Laplacian matrix, while the OM method requires computation of a *generalized* eigenvector (with respect to another Laplacian defining a different metric). This difference between the algorithms accounts for the fact that the OM method pursues rectilinear objects while Normalized Cuts does not.

2.4.3 Selecting Threshold

After having obtained a real-valued solution to (10), we follow previous work (e.g., [15, 11]) and produce a binary segmentation by finding the threshold of the solution x that minimizes (12). Due to the additional boundary term, we have to check both $S = \{i | x_i \geq \tau\}$ and $\bar{S} = \{i | x_i \leq \tau\}$, and report the higher objective value. It should be noted that since this operation takes linear time, the overall time complexity is not affected [11].

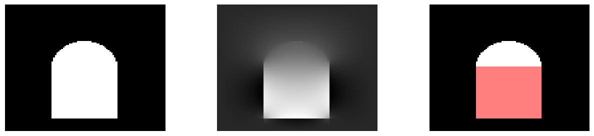
2.4.4 Multiple Rectilinear Object Segmentation

Although the remaining eigenvectors give a solution for K-way segmentation in the case of the normalized cut method [17], it is not possible to recover two overlapping objects with such a device. Since segmentation of overlapping rectangles is highly desirable in our case (e.g., see Section 3.1), we must look beyond the remaining eigenvectors for this purpose. One method of addressing this issue is given by



(a) Original image (b) 1st segmentation (c) 2nd segmentation (d) 3rd segmentation (e) 4th segmentation

Figure 2. Proof of concept. Given the above image, the OM algorithm immediately segments the square without any initialization. Using the device for choosing the next "most rectilinear" object, the objects are chosen in order of how well they optimize (12). This ordering is seen to be: Square, triangle, circle, diamond.



(a) Overlapping circle and square (b) Minimum generalized eigenvector (c) Segmentation (shown in red)

Figure 3. Another example of the algorithm employing the shape preference while also fitting the image data. In this case, a circle and square of the same intensity were joined and the OM algorithm was applied to the image. The result is a decomposition of the object into its constituent parts.

the following device: After computing and discretizing the current segmentation, edges of L_2 are separated according to whether or not they lie on a previous cut. The segmentation is performed a second time on the full image with the following two modifications: 1) The η for each edge group is computed separately, 2) The edge weights in the cut group are additionally multiplied with a decaying constant $0 \leq \nu < 1$. An example of using this device for segmenting multiple rectilinear objects is given in Section 3.1.

2.4.5 Computation

The primary computational burden of this method is the solution of the generalized eigenvectors. The most obvious approach is to find the Cholesky Decomposition of $L_1 + \gamma I_b = A^T A$ (since $(L_1 + \gamma I_b)$ is positive definite), solve for the maximum eigenvector, y, of $\left(A^{-1}\right)^T L_2 A^{-1}$ and then recover the solution for the original problem by solving $Ax = y$.

In order to compute the Cholesky decomposition of matrix $L_1 + \gamma I_b$, we employ the AMD package of [1] to find a permutation of this matrix which is likely to produce a sparser decomposition. After this step, we compute the Cholesky decomposition (see [9]), and solve the resulting sparse eigenvalue problem with ARPACK (see [14]). The mean time required to segment two objects using the device from Section 2.4.4 in MATLAB was 11.2 seconds for the natural images displayed in Section 3. All of these images were of size 100×100.

2.5. Summary

The OM algorithm may be summarized in the following steps:

1. Find weights from the image using (11).
2. Build the L_1 and L_2 matrices as defined by (5) with the geometric weights given by (7) and (8), respectively.
3. Multiply the L_2 weights modified to reflect the image content via (11).
4. Optimize (12) by solving the generalized eigenvector problem $(L_1 + \gamma I_b)\, x = \lambda L_2 x$.
5. Choose a threshold of the resulting eigenvector that minimizes (12) to obtain a final segmentation.

3. Results

In this section, we will begin by showing a proof of concept. As a proof of concept, we show that the algorithm will prefer rectilinear shapes, that it will find a rectilinear shape in the presence of weak or occluded boundaries and that the device for finding multiple rectilinear shapes will work. After establishing proof of concept, we analyze the behavior of the algorithm with regard to resolution (scale) and rotation. Finally, we illustrate the behavior of the algorithm on natural images.

3.1. Proof of concept

The first question that we address is whether or not the algorithm will choose a rectilinear shape from among a collection of objects in an image. Figure 2 shows the result of this experiment. The square is initially chosen by the OM algorithm. Using the above method for finding a K-way segmentation, the other objects are chosen in order of how well they optimize (10). In the case of the image in Figure 2, this ordering is: Square, triangle, circle, diamond.

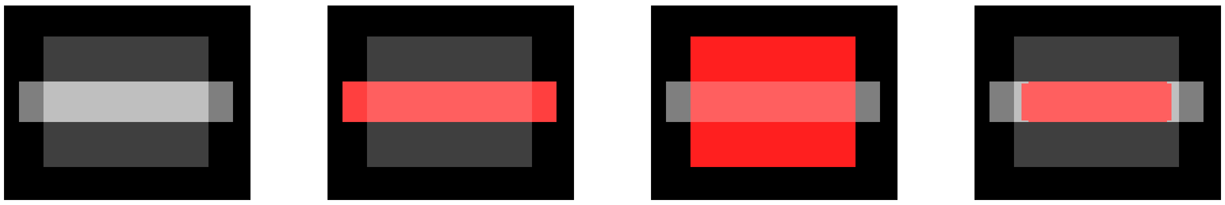

(a) Overlapping rectangles (b) 1^{st} segmentation (c) 2^{nd} segmentation (d) 3^{rd} segmentation

Figure 4. Illustration of the device introduced in Section 2.4.4 to segment multiple (even overlapping) rectilinear objects. Note that the order in which these rectangles are segmented is driven by their relative sizes and contrasts.

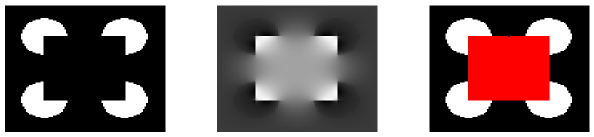

(a) Kanizsa Square (b) Minimum generalized eigenvector (c) Segmentation (shown in red)

Figure 5. The classic demonstration of a prior overriding image data is in the segmentation of the Kaniza examples with illusory boundary. In this figure, we demonstrate that the OM algorithm preference for rectilinear shapes causes it to find the (illusory) square, even though large portions of the boundary are absent.

The proposed segmentation algorithm finds a segmentation that exhibits a tradeoff between image data and a rectilinear shape preference. It is precisely the situation in which the image data is unreliable that the shape prior becomes valuable. We perform two experiments to demonstrate that the OM method will produce the correct segmentations under weak data conditions. The first experiment is to run the OM algorithm on the classic weak-boundary rectilinear shape — The Kaniza square. Figure 5 demonstrates that the OM algorithm will correctly find the Kaniza square, despite the fact that most of the square boundary is completely missing. The second experiment consists of applying the OM algorithm to an object formed by joining a square to a semicircle with the same intensity. Figure 3 demonstrates that the OM algorithm successfully decomposes the image into its constituent parts.

In Section 2.4.4, a method was given for segmenting multiple, overlapping objects in a single image, using the OM approach. In Figure 4 we verify the correctness of this approach.

3.2. Analysis

Shape algorithms are often analyzed from the perspective of translation, scale and rotation invariance of the preferred shape. The translational component is implicitly captured in the OM algorithm, since the graph weights reflect the local intensities, allowing the algorithm to "search" for the best rectilinear shape, regardless of its location in the image.

Our first goal is characterize the dependence of the OM formulation on the scale (resolution) of the rectilinear shape. Scale invariance of any shape algorithm is ultimately limited by the image resolution, since it is impossible to distinguish a one-pixel circle from a one-pixel square, triangle or any other shape. However, due to the combinatorial formulation of our shape criterion (10), it is possible to directly measure how the resolution affects the rectilinearity measure and, consequently, the effect on the segmentation preference. Figure 6 shows the dependence of the ratio $\frac{L_1}{L_2}$ on the resolution of a square for different approximations (neighborhood size) of the L_2 perimeter metric (graph). It may be seen from Figure 6 that there is a weak preference for larger rectilinear shapes, but the strength of this preference is strongly diminished after the square has reached a size of roughly 50–100 pixels on each side. Additionally, we see from Figure 6 that a neighborhood of size three or four is sufficient to approximate the Euclidean (L_2) metric. As expected, an increased neighborhood size and square size drive the metric ratio toward unity.

Rotation invariance is particularly difficult to study on a discretized object, since the rotation operation is much easier to define in the continuum. However, in the present case, it is clear that the measure of rectilinearity that we employ is only minimized when the rectilinear shape is axis-aligned [18]. Since real images are rarely so well behaved, it is important to study the effects of rotation on the algorithm behavior. In order to study the effect of rotation, we designed the following test: We run the object selection test of Figure 2 with the square progressively rotated and determine the point at which the segmentation chooses another object. This "switch point" may be solved for analytically by solving for the rotation of a square that causes a circle to have a smaller objective ratio, as expressed in (10). For a circle of radius r, the perimeter ratio is

$$\frac{\text{Per}_1}{\text{Per}_2} = \frac{8r}{2\pi r} = \frac{4}{\pi}. \tag{13}$$

For a square with side length $2r$ rotated Θ degrees, the

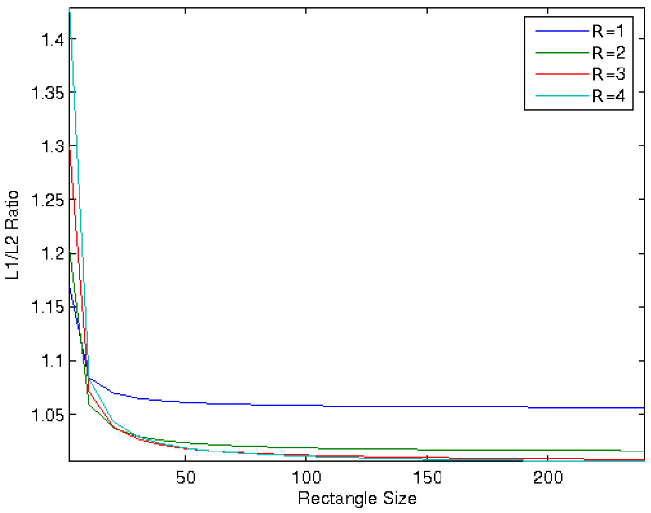


Figure 6. The combinatorial formulation of our shape criteria allows a direct calculation of the effect of resolution (scale) on the shape measure. This plot shows the dependence of the shape measure, $\frac{L1}{L2}$, on Euclidean approximation (neighborhood size) and resolution for squares of increasing size. This experiment demonstrates that it is sufficient to use an L_2 graph to approximate the Euclidean metric by employing neighborhood sizes that connect a pixel to all neighboring pixels at a distance of three or four pixels. Additionally, this experiment also shows that a rectilinear shape consisting of roughly 50–100 pixels is sufficient to drive selection by the algorithm.

perimeter ratio is

$$\frac{\mathrm{Per}_1}{\mathrm{Per}_2} = \frac{8r}{8r(\sin\Theta + \cos\Theta)} = \frac{\sqrt{2}}{2\sin(\Theta + \frac{\pi}{4})}. \quad (14)$$

Equating (13) and (14) yields $\Theta \leq 19°$. Therefore, one would expect that a rotation of 19° degrees would cause the circle to be chosen instead of the rotated rectangle. Experimentally, we created a 100×100 image consisting of a square and circle, and empirically discovered that the circle segmentation is preferred to the rotated square at approximately 17° degrees (with an L_2 graph of neighborhood size two). Therefore, given the image resolution and the ℓ_2 approximation of the L_2 graph, the results generally agree with the prediction from theory (within two degrees). As a result of this experiment we would expect the algorithm to correctly find rectilinear shapes that were roughly axis-aligned, within about 17°. We note that, if strict rotation invariance was essential for a particular application, the above analysis suggests that one could re-apply this algorithm after roughly ten rotations of the image to capture the rotated rectilinear shape. To keep the remaining exposition as simple as possible, such a device will not be employed in the experiments.

3.3. Real Images

The ultimate test of the utility of an algorithm is its ability to operate on natural images. Rectilinear shape segmentation most often arises for applications associated with the segmentation of objects in man-made environments, such as buildings, roadsigns or factory environments. In this section, we demonstrate the response of the OM algorithm to several natural images containing rectilinear shapes. Figure 7 shows the result of this experiment. Note that the objects range in size, contrast and rotation.

4. Conclusion

In this work, we described a new approach for embedding shape information into an image segmentation task. In contrast to previous approaches in the literature, our method produces a global optimum (of the relaxed problem), is non-iterative, steady-state and requires no initialization. Additionally, the algorithm is set in a combinatorial framework. The basis of the Opposing Metrics method is to connect the fact that certain shapes optimize the ratio of perimeters as measured by different metrics [18], with the fact that such metrics may be represented by the weighted connectivity of the graph [3]. The result is a generalized eigenvector problem, which may be solved with standard algorithms. Despite the continuous-valued relaxation of the binary formulation, we have demonstrated that the algorithm behaves in a manner consistent with the motivating formulation. Specifically, the algorithm correctly prefers rectilinear shapes over competing objects and is capable of using the shape preference to "complete" boundaries in the presence of missing, weak or noisy boundaries. Additionally, the device introduced to segment multiple, possibly overlapping, rectilinear shapes behaves as desired.

Future work will be directed at the questions raised by this approach:

1. What classes of shapes may be established as the ratio of their perimeters, as measured by different metrics?

2. Is it possible to employ a combinatorial optimization algorithm to minimize (10) that does not involve a relaxation of the original binary formulation?

3. What properties of a shape may be represented via functions of indicator vectors, such that it is possible to find a steady-state solution? Normalized cuts [15] and the isoperimetric algorithm [11] use the notion of object volume, while we make use here of the measurement of perimeter length via different metrics. Are these the only choices? If not, what other shape properties may be incorporated?

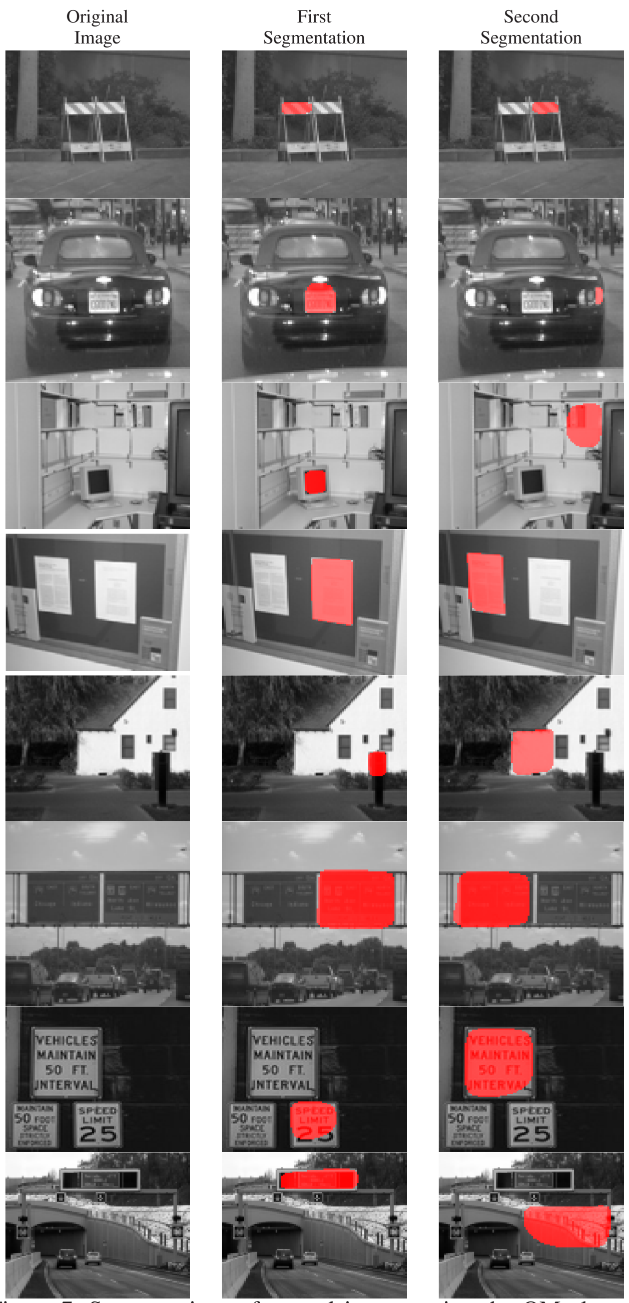


Figure 7. Segmentations of natural images using the OM algorithm. The second segmentation is produced using the device introduced in Section 2.4.4. For all images $\gamma = 1 \times 10^{-2}$.

References

[1] P. Amestoy, T. Davis, and I. Duff. Algorithm 8xx: AMD, an approximate minimum degree ordering algorithm. *ACM Trans. on Math. Software*, 30(3):381–388, Sept. 2004.

[2] Y. Boykov and M.-P. Jolly. *Interactive graph cuts* for optimal boundary & region segmentation of objects in N-D images. In *Proc. of ICCV 2001*, pages 105–112, 2001.

[3] Y. Boykov and V. Kolmogorov. Computing geodesics and minimal surfaces via graph cuts. In *Proc. of ICCV 2003*, volume 1, pages 26–33, Oct. 2003.

[4] T. Cootes, C. Taylor, D. H. Cooper, and J. Graham. Active shape models — Their training and application. *Computer Vision and Image Understanding*, 1995.

[5] T. Cour, F. Benezit, and J. Shi. Spectral segmentation with multiscale graph decomposition. In *Proc. of CVPR 2005*, volume 2, pages 1124–1131, 2005.

[6] D. Cremers. Dynamical statistical shape priors for level set-based tracking. *IEEE Trans. on Pat. Anal. and Mach. Int.*, 28(8):1262–1273, August 2006.

[7] D. Freedman and T. Zhang. Interactive graph cut based segmentation with shape priors. In *Proc. of CVPR 2005*, volume 1, pages 755–762, 2005.

[8] M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of Association for Computing Machinery*, 42:1115 – 1145, 1995.

[9] G. Golub and C. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, 3rd edition, 1996.

[10] L. Grady. Random walks for image segmentation. *IEEE Trans. on Pat. Anal. and Mach. Int.*, 28(11):1768–1783, Nov. 2006.

[11] L. Grady and E. L. Schwartz. Isoperimetric graph partitioning for image segmentation. *IEEE Trans. on Pat. Anal. and Mach. Int.*, 28(3):469–475, March 2006.

[12] J. Keuchel, C. S. C. Schnörr, and D. Cremers. Binary partitioning, perceptual grouping and restoration with semidefinite programming. *IEEE Trans. on Pat. Anal. and Mach. Int.*, 25(11):1364–1379, Nov. 2003.

[13] M. P. Kumar, P. H. S. Torr, and A. Zisserman. OBJ CUT. In *Proc. of CVPR 2005*, volume 1, pages 18–25, 2005.

[14] R. B. Lehoucq, D. C. Sorenson, and C. Yang. *ARPACK User's Guide: Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*. SIAM, 1998.

[15] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. on Pat. Anal. and Mach. Int.*, 22(8):888–905, Aug. 2000.

[16] T. Werner. A linear programming approach to max-sum problem: A review. Technical Report CTU–CMP–2005–25, Center for Machine Perception, Czech Technical University, December 2005.

[17] S. X. Yu and J. Shi. Multiclass spectral clustering. In *Proc. of ICCV 2003*, volume 1, pages 313–319, Nice, Oct. 2003.

[18] J. Zunic and P. Rosin. Rectilinearity measurements for polygons. *IEEE Trans. on Pat. Anal. and Mach. Int.*, 25(9):1193–1200, Sept. 2003.