Segmentation by transduction

Olivier Duchenne
Willow - ENS / INRIA
45 rue d'Ulm
75005 Paris - France
olivier.duchenne@ens.fr

Jean-Yves Audibert
CERTIS - Ecole des Ponts
Willow - ENS / INRIA
audibert@certis.enpc.fr

Renaud Keriven
CERTIS - Ecole des Ponts
6 avenue Blaise Pascal - Cité Descartes
77455 Marne-la-Vallée - France
keriven@certis.enpc.fr

Jean Ponce
Willow - ENS / INRIA
ponce@di.ens.fr

Florent Ségonne
CERTIS - Ecole des Ponts

Abstract

This paper addresses the problem of segmenting an image into regions consistent with user-supplied seeds (e.g., a sparse set of broad brush strokes). We view this task as a statistical transductive inference, in which some pixels are already associated with given zones and the remaining ones need to be classified. Our method relies on the Laplacian graph regularizer, a powerful manifold learning tool that is based on the estimation of variants of the Laplace-Beltrami operator and is tightly related to diffusion processes. Segmentation is modeled as the task of finding matting coefficients for unclassified pixels given known matting coefficients for seed pixels. The proposed algorithm essentially relies on a high margin assumption in the space of pixel characteristics. It is simple, fast, and accurate, as demonstrated by qualitative results on natural images and a quantitative comparison with state-of-the-art methods on the Microsoft GrabCut segmentation database.

1. Introduction

Image segmentation, the process of partitioning an image into "meaningful" regions (Figure 1), is a fundamental task in a large number of applications in computer vision, medical imaging, etc. For instance, it may be used to separate an object from its background (e.g., identification of specific anatomical structures in medical images, tracking of persons or objects in video sequences, etc.), or identify image areas pertinent to some application (e.g., forests, fields, or towns in satellite imagery). Segmentation also has obvious applications in painting software (alpha matting for landscape recomposition, virtual reality, etc.). More generally, effective segmentation methods are a key to better scene understanding and object recognition.

Figure 1. Left: an input image with user-supplied strokes. Right: the segmentation found by the algorithm proposed in this paper.

Yet, current segmentation algorithms are far from matching human performance for natural images. The difficulty of the task at hand and the limitations of the input data (real images differ from their models because of noise, occlusion, clutter, etc.) often lead to ill-posed problems. The segmentation process itself is in general ill-defined without additional prior knowledge: Homogeneity in some a priori feature space such as, say, color or texture, is not a sufficient criterion to define a meaningful, unambiguous image partition. People probably rely on a combination of low-level information (e.g., color, texture, contours) and high-level knowledge (e.g., shape priors, semantic cues) to resolve these ambiguities. On the other hand, the incorporation of prior knowledge in automated segmentation techniques is still a challenging and active research area.

The literature dedicated to knowledge-driven image segmentation is abundant. It includes completely automated methods [8, 10, 22, 25], semi-supervised methods [4, 12, 11, 9, 13, 17, 23, 1], and model-driven approaches [18, 20, 26], among many others. Semi-supervised (or *interactive*) segmentation methods classify unlabelled data (i.e., segment unknown regions) provided information in the form of

978-1-4244-2243-2/08/$25.00 ©2008 IEEE

labelled and unlabelled training data (i.e., some segmented regions, usually user-defined). Since we know beforehand the points (or pixels) to be labelled, the problem is one of *transductive learning*. This is the viewpoint adopted in this presentation, and we define our segmentation problem as follows:

Segmentation by transduction: *Given a set of user-supplied seeds representative of each region to be segmented in an image, generate a segmentation of the entire image that is consistent with the seeds.*

The first innovation of this paper is to explicitly cast the segmentation problem in a transductive learning framework. We make three other contributions: First, we introduce the s-weighted graph Laplacian regularizer to solve the transduction problem. It appears that several segmentation methods can be re-interpreted as special cases of our general framework, and we highlight the corresponding connections. Second, we clearly illustrate the link between the continuous formulation of transductive inference and its discrete counterpart, introducing a "free" parameter $\lambda = 1 - s/2$ as a measure of the output variation on low-density input regions. As a consequence, our discrete formulation leads to an energy minimization which reduces to solving a linear system of size proportional to the number of pixels to be labelled, resulting in a simple and fast segmentation procedure. Third, and finally, we demonstrate with qualitative and quantitative experiments on natural images that our segmentation algorithm is also accurate (see Figures 1 and 3 for typical examples). In particular, a quantitative evaluation shows that our method compares favorably with other state-of-the-art approaches [4, 12, 11, 13] on the Microsoft GrabCut segmentation dataset [4].

The rest of our presentation is organized as follows. Section 2 presents our transductive view of image segmentation, and Section 3 describes our segmentation algorithm. Section 4 discusses its relationship with previous research. Section 5 presents our experimental results, and we conclude in Section 6 with a brief discussion.

2. A transductive view of image segmentation

2.1. Transductive vs Inductive inference

One of the main issues addressed by machine learning is labeling of new data points given a set of labelled examples; classically, one observes input-output pairs and wants to derive from this training data the outputs associated with new inputs. One should distinguish transductive from inductive inference: In inductive inference, the new inputs are not known beforehand, so that the algorithm has to learn from the training set a mapping from the whole input space to the output space. When new (test) inputs come in, the learned function maps them to corresponding outputs.

In transductive inference, the setup is different: The training set and the *input* test set are both given from the start (the *output* test set is of course hidden). As a consequence, the two-step process of learning the input-to-output mapping before using it on new test points can be replaced by a single one: learning the output associated with the input test points. This methodology follows Vapnik's principle: *Do not try to learn more than necessary as an intermediate step*. It is the one often adopted *implicitly* in so-called *semi-supervised* approaches to segmentation [4, 9, 13, 17]. Note that, strictly speaking, semi-supervised tasks [28] are slightly different to transductive learning since instead of starting with a labelled training set and a set of input test points, semi-supervised learning methods start with a labelled and unlabelled training set and do not know in advance the points to be classified.

In both transductive and semi-supervised settings, one can use unlabelled points to get an idea of the input distribution, which often turns out to be useful, as illustrated by a second key principle, often referred to in the machine learning community as the clustering assumption: *Outputs only vary a lot in regions of the input space having low density*.

Figure 2 points out some of the advantages of transductive classifiers compared to inductive ones: In this toy example, we have to differentiate two families of points in $\mathbb{R}^2$. The learning sets are represented in the top left part of the figure. An inductive classifier would find a separator in the middle of the two classes (bottom left). Thanks to the unlabelled points (top right), a transductive method will find a separator in a low-density area, resulting in a much more satisfactory contour (bottom right).

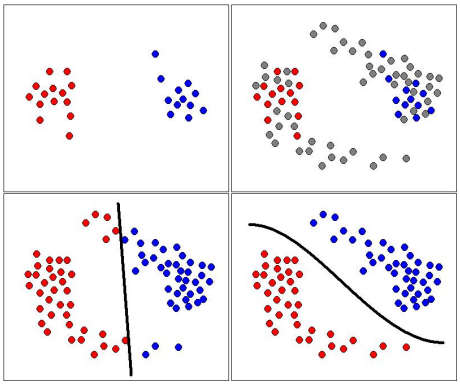

Figure 2. Top left: the training data. Top right: the unlabelled points in gray. Bottom left: the separator found by a (hypothetical) inductive algorithm, and the corresponding results. Bottom right: the separator fround by a transductive algorithm and the corresponding results. The presence of unlabelled points is used to put the boundary in low density region of the input space. This is specially useful when only few points are labelled.

In this paper, we explicitly cast segmentation in a transductive framework, which enables us to introduce the s-weighted Laplacian graph regularizer, a powerful manifold learning tool that is based on the estimation of variants of the Laplace-Beltrami operator and is tightly related to diffusion processes.

2.2. The segmentation input space

In our multi-zone segmentation problem, the outputs associated with seed pixels are given; the inference problem consists in determining the outputs associated with the remaining pixels of the image. This is basically a transductive classification task in which the number of classes is the number of selected zones in the segmentation. In machine learning, the question of representation is of crucial importance. This is the first one we address: How should we represent image pixels? Our choice for the input space is motivated by the following considerations:

• pixels coming from the same zone should be well clustered;

• clusters coming from different zones should be well separated;

• geometric (position) as well as photometric (color, texture) information should be used to cluster pixels.

We use very simple features in our implementation: we capture the photometric information at each pixel by recording the image colors in a fixed-size window surrounding it, and encode geometric information in the form of pixel coordinates. More sophisticated encodings of texture and geometry could of course have been used as well (e.g., [16, 27]).

In order to capture the *texture* of the different objects, we associate with each pixel a local patch centered around it: the texture is then encoded by the color level of the patch. Finally, the geometric information is just encoded by the row and column numbers of the pixel. It is interesting to note that while the combined use of color and texture is standard, geometric position has been less employed despite its crucial importance in the image segmentation problem.

The pixel position information is really important as one can see by considering images in which two identical objects should be put in two different zones.

2.3. The graph Laplacian method: a state-of-the-art transductive inference algorithm

Methods based on graph Laplacians have emerged recently and have been successfully used in transductive inference [3], (spectral) clustering [24], and dimensionality reduction [2].

The underlying assumption in these methods is that *the (input) points are generated by a probability distribution with support on a submanifold of the Euclidean space*. Let M denote this submanifold, and p denote the density of the input probability distribution with respect to the canonical measure on M. When M is the Euclidean space itself (or a manifold with the same dimension), p can simply be viewed as a density with respect to the Lebesgue measure on that space.

In transductive inference, one searches for a smooth function f from the input space into the output space such that $f(X_i)$ is close to the associated output Y_i on the training set, and such that the function is allowed to vary only on low-density regions of the input space. Let $s \geq 0$ be a parameter characterizing how low the density should be to allow large variations of f, i.e., we consider a s-weighted version of the density p. *One of the main contributions of this work consists in introducing the "free" parameter s and in carefully explaining its connection with a specific graph Laplacian. We also suggest how to choose the parameter s.*

For the sake of clarity, let us consider a real-valued output space (such as the space of alpha-matting coefficients in a two-zone segmentation task [21]). Depending on the confidence we assign to the training outputs, we obtain the following optimization problem:

$$\min_f \sum_{i \in T} c_i [Y_i - f(X_i)]^2 + \int_M \|\nabla f\|^2 p^s dV, \qquad (1)$$

where the summation is over the pixels of the training set T, the c_i's are positive coefficients measuring how much we want to fit the training point (X_i, Y_i), and the integral term essentially forces the function f to vary only in low-density regions. Typically, $c_i = +\infty$ imposes a hard constraint on the function f so that $f(X_i) = Y_i$. The *s-weighted Laplacian operator* is characterized by:

$$\int_M f \times (\Delta_s g)\, p^s dV = \int_M \langle \nabla f, \nabla g \rangle\, p^s dV, \qquad (2)$$

where f and g are smooth real-valued functions defined on M with compact support. By the law of large numbers and using (2), the integral in Eq. (1) can be approximated by

$$\tfrac{1}{n} \textstyle\sum_{i=1}^n f(X_i) \Delta_s f(X_i) p^{s-1}(X_i).$$

Unfortunately, the direct computation of $\Delta_s f(X_i)$ for every possible function f is not possible and solving Eq. (1) is intractable.

As shown by Hein *et al.* in [14], *graph Laplacian methods, which are based on a discrete approximation of the s-weighted Laplacian operator, propose a discrete alternative to this problem.* Briefly, these methods are based on a neighborhood graph in which the nodes are the input points coming from both the training and test sets. Let $X_1, \ldots, X_n$ denote these points. Let $\tilde{k} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a symmetrical function giving the similarity between two input points. The typical kernel $\tilde{k}$ is the Gaussian kernel $\tilde{k}(x', x'') = e^{-\frac{\|x' - x''\|^2}{2h^2}}$, where $h > 0$ is commonly referred to as the bandwith of the kernel. The degree function associated with this first kernel is defined by $\tilde{d}(x) = \sum_{i=1}^n \tilde{k}(X_i, x)$. Given some scalar $\lambda \geq 0$, general graph Laplacian methods use the normalized kernel defined as

$$k(x', x'') = \frac{\tilde{k}(x', x'')}{[\tilde{d}(x') \tilde{d}(x'')]^\lambda}. \qquad (3)$$

For $\lambda = 0$, no normalization is done. For $\lambda = 1/2$, the normalization is perfect to the extent that multiplying $\tilde{k}$ by a constant does not change the value of k. The degree function associated with this second kernel is $d(x) = \sum_{i=1}^n k(X_i, x)$.

The kernel k induces a weighted undirected graph in which the nodes are $X_1, \ldots, X_n$, and any two nodes are

linked with an edge of weight $k(X_i, X_j)$. The degree of a node is defined by the sum of the weights of the edges at the node, i.e., $d(X_j)$.

Let us define W as the $n \times n$ matrix such that $W_{ij} = k(X_i, X_j)$, D as the diagonal $n \times n$ matrix such that $D_{ii} = d(X_i)$, and I as the $n \times n$ identity matrix. As shown in [14], three kinds of graph Laplacian can be defined in terms of these matrices:

— the random walk matrix: $L_{\text{rw}} = I - D^{-1}W$;
— the unnormalized matrix: $L_{\text{un}} = D - W$;
— the normalized matrix: $L_{\text{n}} = I - D^{-1/2}WD^{-1/2}$.

For a given function $f : \mathcal{X} \to \mathbb{R}$, let F be the vector defined as $F_i = f(X_i)$. The main result of [14] is essentially that

$$\left\{ \begin{array}{l} (L_{\text{rw}}F)_i \rightsquigarrow \big(\Delta_{2(1-\lambda)}f\big)(X_i), \\ (L_{\text{un}}F)_i \rightsquigarrow [p(X_i)]^{1-2\lambda}\big(\Delta_{2(1-\lambda)}f\big)(X_i), \\ (L_{\text{n}}F)_i \rightsquigarrow [p(X_i)]^{\frac{1}{2}-\lambda}\Big[\Delta_{2(1-\lambda)}\big(\frac{f}{p^{1/2-\lambda}}\big)\Big](X_i), \end{array} \right. \quad (4)$$

where $\rightsquigarrow$ means almost sure convergence when the sample size n goes to infinity and the kernel bandwidth h goes to zero not too rapidly (e.g., $h = (\log n)^{-1}$), up to normalization of the left-hand side by an appropriate function of n and h. In addition, one can understand the role of the degree functions through the convergences:

$$\left\{ \begin{array}{l} \tilde{d}(x) \rightsquigarrow p(x), \\ d(x) \rightsquigarrow [p(x)]^{1-2\lambda}. \end{array} \right.$$

For instance, for $\lambda = 1/2$, the three graph Laplacians are essentially the same since the matrix D converges to the identity matrix.

The above analysis suggests that instead of focusing on the intractable optimization (1), one may want to solve a simple quadratic problem (possibly with linear equality constraints if some c_i's are infinite), namely:

$$\min_{F \in \mathbb{R}^n} \sum_{i \in T} c_i(Y_i - F_i)^2 + F^t L_{\text{un}} F, \quad (5)$$

where the summation is once again taken over the training pixels T, and $\lambda = 1 - s/2$. The parameter $\lambda \leq 1$ (or equivalently the parameter $s \geq 0$) exerts an influence on the segmentation by characterizing how low the density should be to allow large variations of the labeling function. Larger values of s (i.e., $s \geq 1$) should reflect the user's belief that the labelled data are well-clustered in the feature space.

Let C be the diagonal $n \times n$ matrix for which the i-th diagonal element is c_i for a training point, and 0 for a test point. Similarly let Y be the n-dimensional vector for which the i-th element is Y_i for a training point, and 0 for a test point. Equation (5) reduces to

$$\min_{F \in \mathbb{R}^n} (F - Y)^t C(F - Y) + F^t L_{\text{un}} F, \quad (6)$$

whose solutions satisfy the linear system

$$(L_{\text{un}} + C)F = CY. \quad (7)$$

For infinite coefficients c_i, we have $F_i = Y_i$ on the training set. Now by setting the gradient of $F^t L_{\text{un}} F$ to zero, we solve the minimization problem (6) and obtain that the F_j's on the input test set J are the solutions of the system:

$$\textstyle\sum_{j \in J} L_{k,j} F_j = -\sum_{i \in \{\text{train pixels}\}} L_{k,i} Y_i, \quad (8)$$

for all rows k of the matrix L.

3. Proposed segmentation algorithm

3.1. Two-zone segmentation

Our segmentation algorithm is parameterized by:
— $s \geq 0$: a measure of how much we believe that "outputs should vary only on input regions having low density" (see Eq. (1));
— $\sigma_g > 0$: scale of geometric neighbourhoods (see Eq. (9));
— $\sigma_c > 0$: scale of chromatic neighbourhoods (see Eq. (9));
— $m \in \mathbb{N}$: size of the local patch (see below).

Let $C(i)$ denote the RGB levels of a square patch of size $2m + 1$ around the pixel i. Let x_i denote the geometric position (row+column) of the pixel i. We use the following kernel between pixels

$$\tilde{k}(i, j) = e^{-\frac{\|x_i - x_j\|^2}{2\sigma_g^2} - \frac{\|C(i) - C(j)\|^2}{2\sigma_c^2}}. \quad (9)$$

The labels of the training pixels are either 0 or 1 depending which zone the pixel i belongs to. Finally, our segmentation method consists in:
(1) computing L_{un} (see (4); it uses (3) with $\lambda = 1 - s/2$)
(2) solving the sparse linear system (8)
(3) thresholding the output to $1/2$: the pixel j is assigned to zone $\mathbf{1}_{F_j \geq 1/2}$.

3.2. Multi-zone segmentation

The previous procedure can easily be extended to a multi-zone segmentation with more than two zones. Let d denote the number of zones. The output value associated with the k-th zone is defined to be the vector of $\mathbb{R}^d$ having all zero coefficients except its k-th coefficient being equal to one. Then, in order to produce a smooth function that is coherent with the observed outputs on the training points (especially on high-density regions), we solve (8) for each coordinate f_k, $k = 1, \ldots, d$. This procedure can be viewed as a simple one-vs-all segmentation method; at the end, a pixel is assigned to the zone l, where $l = \text{argmax}_{k=1,\ldots,d} f_k$.

3.3. Segmentation with prior knowledge of the zones

One might want to use the algorithm without any user-provided seeds. Consider a two-zone segmentation problem

(object vs background): If no information is given on the zones, then one can use the prior proposed in [13, Section 2.2]. In some situations, on the other hand, the user has prior knowledge of the following form: Each pixel i has a score s_i measuring the likelihood that the pixel belongs to the object zone. This is in particular the case when the user wants to segment an object with a known texture, previously learned from a database of objects of the same category. This information can be directly plugged into the segmentation method by adding the term: $c\sum_{i\in\{\text{test pixels}\}}(s_i - F_i)^2$. The computational complexity of the method as characterized below remains unchanged.

3.4. Computational complexity

The main computational cost of the algorithm comes from solving the linear system (8). Using a truncated version of the Gaussian kernel, the resulting $n \times n$ matrix becomes sparse. Solving such a system can be done in $O(n.p)$, where n is the size of the square matrix (the number of unlabelled points) and p the number of non-zero entries in the matrix (the total number of neighbors for all the points). In comparison, graph-cut algorithms which are known to be fast use the Ford-Fulkerson min-cut max-flow algorithm. Its complexity is $O(E.f)$, where E is the number of edges and f the maximum flow. In our case, E is equivalent to p and f is proportional to n, showing that the two algorithms have the same complexity.

4. Links with previous approaches

4.1. Graph cuts

When the labels Y_i's take their value in $\{-1; +1\}$, a variant of our segmentation algorithm would be to replace (5) with $\min_{F\in\{-1;1\}^n} \sum_{i\in T} c_i(Y_i - F_i)^2 + F^t L_{\text{un}} F$ so that steps (2) and (3) are replaced with the combinatorial task

$$\min_{\substack{F\in\{-1;1\}^n \\ F_i = Y_i \text{ on } T}} F^t L_{\text{un}} F. \tag{10}$$

This problem can be efficiently solved by a graph-cut algorithm. Straightforward computations show that the edge between two pixels i and j should be weighted by four times the (i, j)-element of the matrix W.

For $s = 1$ (equivalently $\lambda = 1/2$ in (3)), the matrix which appears is exactly the one of the normalized cut eigenvalue problem (see [22]). Besides, the discrete version of our algorithm for $s = 2$ (equivalently $\lambda = 0$ in (3)) is comparable to the regularization used in [4, 5, 6].

4.2. Iterative segmentation methods

Several segmentation procedures (e.g., using level sets [20] or iterated graph cuts [19]) rely on the evolution of a curve or region in which, at each step, the visual properties (e.g., color and texture) of the regions are updated. This leads to generally slow methods because of the iterative aspects of the problem. We believe that our viewpoint is conceptually and practically more satisfactory since the "chicken-and-egg" aspect is directly encoded in our global optimization problem.

4.3. The random walker

In [12, 11], Grady and Funka-Lea motivate their algorithm by interpreting the similarities between pixels in terms of transition probabilities from one pixel to another (the ones given by L_{rw}). Then, for a given pixel, they consider (infinitely many) random walkers starting from this pixel and all moving according to these transition probabilities. The motivating idea of their algorithms is to assign a pixel to the class having received the largest fraction of random walkers. Interestingly, their implementation does not use the random walk Laplacian L_{rw}. It rather uses the unnormalized Laplacian L_{un} (motivated in their work by potential theory in electrical circuits), so that their algorithm corresponds to our method for the parameter $s = 2$.

4.4. Guan and Qiu's approach

Our framework is capable of handling the energy underlying Guan and Qiu's approach [13]. The minimized energy functional is the following:

$$\min_{F\in\mathbb{R}^n} \sum_{i\in T} c_i(Y_i - F_i)^2 + F^t L_{\text{rw}}^2 F, \tag{11}$$

where the summation is as usual done on the training set, with $c_i = +\infty$ and $\lambda = 0$. In other words, they solve the discrete version of

$$\min_f \int_M |\Delta f|^2 p dV, \tag{12}$$

where f is constrained to verify $Y_i = f(X_i)$ on T.

Considering the regularizer $\int_M \|\nabla f\|^2 p dV$ (and its variants) seems to us more adequate than the above regularizer since the former gives no penalty for linearly-varying functions.

5. Experimental results

Figures 1 and 3 show several segmentation examples on natural images, where the user input is limited to a small set of broad brush strokes. The results are qualitatively good, and mostly agree with perceptual boundaries.

For quantitative results, we turn to the Microsoft GrabCut database [4].[1] Despite the availability of numerous datasets with ground truth segmentations, it is, to the best of our knowledge, the only one for which seed regions are given. It contains, however, seeds of a very particular type since all pixels are labelled except for a narrow band around the contour of the segmented object. Nevertheless, it is the

[1] http://research.microsoft.com/vision/cambridge/i3l/segmentation/GrabCut.htm

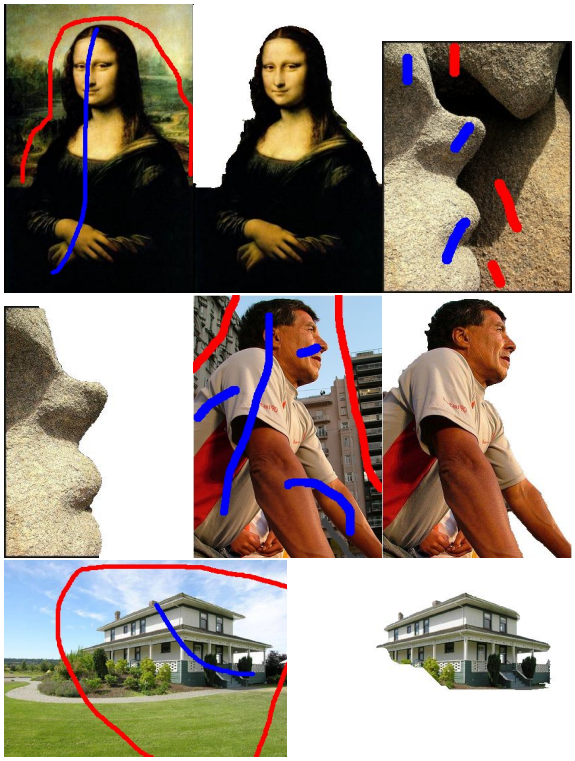

Figure 3. Qualitative segmentation results in the same format as Figure 1.

Segmentation model	Error rate
GMMRF ([4])	7.9%
Random Walker (s=2) ([11])	5.4%
Our method without AT	5.4%
Square Laplacian regularizer ([13])	4.6%
Random Walker (s=2) with AT	3.3%
Our method with AT	3.3%

Table 1. Pourcentage of mislabelled pixels in the region to be classified. Note that the two last scores correspond to algorithms dedicated to segmentation with contour information (using an adaptative filter [13]). A value of $s = 1.5$ has been used in these experiments.

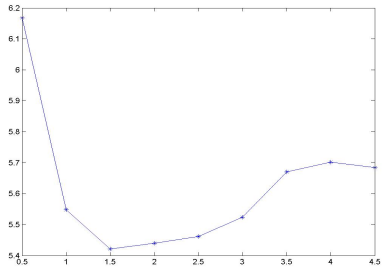

Figure 5. Influence of the parameter s on the average error on Microsoft database

only database which permits us to give a quantitative comparison with state-of-the-art algorithms.

It may be argued that this dataset is of limited interest since one can exploit the particular form of the seeds to provide a good segmentation. Indeed, a naive segmentation approach that would track the skeleton of the unlabelled points might perform quite well on this data set. The adaptative thresholding (AT) method introduced by Guan and Qui in [13, Section 3.1] essentially produces the same effect. Using the same adaptive filter, we have also greatly improved our segmentation results, leading to significantly better results than the ones reported in [13]. To distinguish the actual capabilities of our method from the appropriateness of the AT step to a particular dataset, we have chosen to show our results with and without that step in the comparative evaluation of Table 1.

Following [4], we believe that, even if the masks considered in the database contain a considerable amount of information, the interactive segmentation task illustrated by this data is still of considerable practical interest. For instance, it is easy for a user to use a broad brush and provide the band where the boundary of the object has to be searched for. Besides, several object detection algorithms provide a rough estimate of the region in which the object is (see e.g., [7, 15]), and in most cases, this information can easily be converted into a band containing the boundary.

Figure 4 illustrates some typical outputs of our segmentation with adaptative thresholding (AT) and $s = 1.5$. The last column, which corresponds to the worst score (i.e., 9.15%) of our segmentation algorithm is still of great quality.

5.1. Discussion

5.1.1 Geometric neighborhood

In the experiments described in this paper, we have considered a relatively small geometric neighborhood ($\sigma_g = 10$), a small chromatic neighborhood ($\sigma_c = 10$) and the minimal patch size ($m = 1$). We have found empirically that the graph Laplacian is not efficient with a too large σ_g. Isolated pixels appear, large zones influence become too important. So the algorithm cannot use long range pixel similarity. To tackle this problem, we think that it shoud be used in association with long-range or global methods such as a Gaussian mixture color model [4] or a SVM classifier. This could be done using the method explained in Section 3.3. The graph Laplacian would help to diffuse the prior knowledge of a smart classifier.

5.1.2 Comparison with graph cuts

Algorithms based on graph Laplacians and graph cuts can minimize the same energy function by an appropriate change of kernel. They have the same theoretical complex-

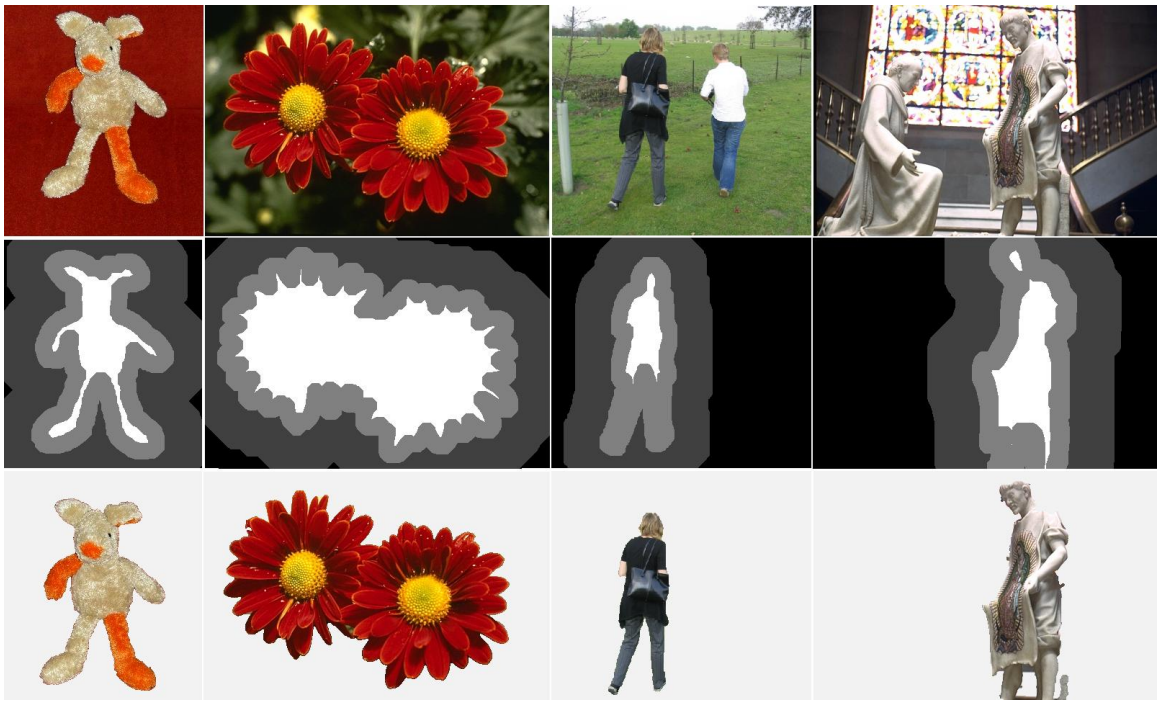

Figure 4. Top row: Initial images. Middle row: Masks provided for the segmentation. Bottom row: Results of our segmentation with $s = 1.5$. Our method performs essentially as well as the case $s = 2$ ([11]; "random walker") with a score of 3.3%. The last column corresponds to the worst segmentation with 9.15%.

ity but it is well known that, in many computer vision tasks, the running time of graph cuts is below what is expected from its theoretical complexity. However, graph cuts provide only labels and our algorithm provide a real-valued score function. This score function can be used to perform alpha matting, or it could be used as a confidence score. Indeed zones different from both learning zones received an almost null score.

5.1.3 Computational time

The segmentation process takes between two seconds and three minutes on a Pentium 1.7 GHz for the database images. Real time is not yet reachable. But this time could be improved by standard multicore methods for sparse system.

5.1.4 Influence of the "free" parameter s

At this point, our evaluation of the effect of the parameter s on segmentation performance remains quite preliminary. A quantitative evaluation using the Microsoft GrabCut database only shows a modest influence of its value on the segmentation error rate (less than 1% in the operating range of our experiments, with an optimal value around $s = 1.5$, see Fig. 5). To get a deeper understanding of these effects, we are conducting a series of simulations on synthetic examples. So far, these experiments suggest that when s gets very large, the influence of the locally densest training points becomes so important that neighbor test points get labelled exactly the same way. On the other hand, when s gets smaller, the f function becomes very smooth.

6. Conclusion

We have presented a simple, yet accurate segmentation procedure in a transductive framework. We clearly illustrated the link between the continuous formulation and its discrete counterpart, introducing a parameter s as a measure of the output variations on low-density input regions. Our discrete formulation leads to an energy minimization which reduces to a linear system of size proportional to the number of pixels to be labelled. Segmentation results on natural images clearly demonstrate the quality of our approach, and a quantitative evaluation on the data set of [4] has shown that our method performs very well (the results are essentially similar to the ones in [11] which corresponds to $s = 2$). Future work will focus on improving the design of the kernel to make it better adapted to the segmentation task at hand, and on reducing the complexity of the algorithm and making it real-time in high-resolution images.

Acknowledgments

This work was supported in part by the National Science Foundation under grant IIS-0535152 and by the Agence Nationale de la Recherche project "Modèles Graphiques et Applications".

References

[1] X. Bai and G. Sapiro. A geodesic framework for fast interactive image and video segmentation and matting. In *IEEE 11th International Conference on Computer Vision*, 2007.

[2] Belkin and Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Comp.*, 15(6):1373–1396, 2003.

[3] Belkin and Niyogi. Semi-supervised learning on manifolds. *Machine Learning*, 56:209–239, 2004.

[4] A. Blake, C. Rother, M. Brown, P. Perez, and P. Torr. Interactive image segmentation using an adaptive GMMRF model. In *ECCV06*, pages I: 428–441, 2006.

[5] Y. Boykov and G. Funka-Lea. Graph cuts and efficient n-d image segmentation. *International Journal of Computer Vision (IJCV)*, 70(2):109–131, 2006.

[6] Y. Boykov and M.-P. Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In *International Conference on Computer Vision (ICCV)*, volume I, pages 105–112, 2001.

[7] B. Epshtein and S. Ullman. Identifying semantically equivalent object fragments. In *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1*, pages 2–9, Washington, DC, USA, 2005. IEEE Computer Society.

[8] F. J. Estrada and A. D. Jepson. Quantitative evaluation of a novel image segmentation algorithm. In *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2*, pages 1132–1139, Washington, DC, USA, 2005. IEEE Computer Society.

[9] M. Figueiredo, D. S. Cheng, and V. Murino. Clustering under prior knowledge with application to image segmentation. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*. MIT Press, Cambridge, MA, 2007.

[10] M. Galun, E. Sharon, R. Basri, and A. Brandt. Texture segmentation by multiscale aggregation of filter responses and shape elements. In *ICCV '03: Proceedings of the Ninth IEEE International Conference on Computer Vision*, page 716, Washington, DC, USA, 2003. IEEE Computer Society.

[11] L. Grady. Random walks for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(11), 2006.

[12] L. Grady and G. Funka-Lea. Multi-label image segmentation for medical applications based on graph-theoretic electrical potentials. In M. Sonka, I. A. Kakadiaris, and J. Kybic, editors, *ECCV Workshops CVAMIA and MMBIA*, volume 3117 of *Lecture Notes in Computer Science*, pages 230–245. Springer, 2004.

[13] J. Guan and G. Qiu. Interactive image segmentation using optimization with statistical priors. In *International Workshop on The Representation and Use of Prior Knowledge in Vision, In conjunction with ECCV 2006, Graz, Austria*, 2006.

[14] M. Hein, J.-Y. Audibert, and U. von Luxburg. From graphs to manifolds - weak and strong pointwise consistency of graph Laplacians. ArXiv Preprint, Journal of Machine Learning Research, forthcoming, 2006.

[15] M. P. Kumar, P. H. S. Torr, and A. Zisserman. Obj cut. In *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1*, pages 18–25, Washington, DC, USA, 2005. IEEE Computer Society.

[16] S. Lazebnik, C. Schmid, and J. Ponce. A sparse texture representation using local affine regions. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 27(8):1265–1278, August 2005.

[17] B. Micusik and A. Hanbury. Automatic image segmentation by positioning a seed. In *ECCV'06*, pages II: 468–480, 2006.

[18] N. Paragios and R. Deriche. Geodesic active regions and level set methods for supervised texture segmentation. *Int. J. Comput. Vision*, 46(3):223–247, 2002.

[19] C. Rother, V. Kolmogorov, and B. A. Grabcut - interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics, SIGGRAPH 2004*, 2004.

[20] M. Rousson, T. Brox, and R. Deriche. Active unsupervised texture segmentation on a diffusion based space. In *International Conference on Computer Vision and Pattern Recognition*, volume 2, pages 699–704, Madison, Wisconsin, USA, June 2003.

[21] M. Ruzon and C. Tomasi. Alpha estimation in natural images. In *Computer Vision and Pattern Recognition*, pages 18–25, 2000.

[22] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[23] A. Sinop and L. Grady. A seeded image segmentation framework unifying graph cuts and random walker which yields a new algorithm. In *IEEE 11th International Conference on Computer Vision*, pages 1–8, 2007.

[24] D. Spielman and S. Teng. Spectral partioning works: planar graphs and finite element meshes. In *37th Ann. Symp. on Found. of Comp. Science (FOCS)*, pages 96–105. IEEE Comp. Soc. Press., 1996.

[25] D. A. Tolliver and G. L. Miller. Graph partitioning by spectral rounding: Applications in image segmentation and clustering. In *CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1053–1060, Washington, DC, USA, 2006. IEEE Computer Society.

[26] A. Tsai, A. Yezzi, W. Wells, C. Tempany, D. Tucker, A. Fan, W. Grimson, and A. Willsky. A shape-based approach to the segmentation of medical imagery using level sets. *IEEE Transactions on Medical Imaging*, 22(2):137–154, 2003.

[27] M. Varma and A. Zisserman. Texture classification: Are filter banks necessary? In *CVPR '03: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'03) - Volume 2*, pages 691–698, 2003.

[28] X. Zhu. Semi-supervised learning literature survey. Technical Report TR–1530, 2005.