8 Continuous-Valued MRF for Image Segmentation

Dheeraj Singaraju, Leo Grady, Ali Kemal Sinop, and René Vidal

Research on image segmentation has focused on algorithms that *automatically* determine how to group pixels into different regions on the basis of homogeneity of intensity, color, texture, or other features [434, 100, 138, 533]. However, since images generally contain several objects that are surrounded by clutter, it is often not possible to define a unique segmentation. In many such cases, different users may be interested in obtaining different segmentations of an image. Hence, recent research in segmentation has focused on *interactive* methods that allow different users to interact with the system and to segment different objects of interest from the same image.

One genre of interactive segmentation algorithms offers the user a *scribble interface* to label two disjoint sets of pixels as belonging to the object of interest and to the background. The algorithms' goal is then to output a label for each unmarked pixel into one of these two categories. The labeling is typically obtained by minimizing an energy function defined on a weighted combinatorial graph. In general, this can be done using several methods, such as graph cut [66, 72], random walker [175], shortest path [18, 113], region growing [2], fuzzy connectivity [487], seeded watershed [24], and many more examples given in chapter 7. This genre has become very popular, notably due to the availability of numerical solvers that efficiently produce the global optimizer of the defined energy function.

This chapter discusses a generalized graph-theoretic algorithm that estimates the segmentation via a continuous-valued optimization as opposed to the traditional view of segmentation as a discrete-valued optimization, as in chapter 7. The algorithm proceeds by associating a continuous-valued variable with each node in the graph. An energy function is then defined by considering the p-norm of the difference between these variables at neighboring nodes. The minimizer of this energy function is thresholded to produce a binary segmentation.

This formulation includes algorithms such as graph cut [66], random walker [175], and shortest path [18] as special cases for specific values of the p-norm (i.e., $p = 1, 2$, and ∞, respectively). Due to the choices of the p-norm, these algorithms have their characteristic disadvantages. Three such concerns that will be discussed in detail later are *metrication artifacts* (blockiness of the segmentation due to the underlying grid structure), *proximity bias* (bleeding of the segmentation due to sensitivity to the location of user interaction),

and *shrinking bias* (shortcutting of the segmentation boundary due to bias toward shorter boundaries).

The use of an intermediate p-norm for segmentation might compensate for these drawbacks. However, the optimization of intermediate p-norms has been somewhat neglected, due to the focus on fast dedicated solvers for the cases of $p = 1$, $p = 2$, and $p = \infty$ (e.g., max-flow for $p = 1$, linear system solver for $p = 2$, and Dijkstra's shortest path algorithm for $p = \infty$). The lack of a general solver precludes the ability to merge these algorithms or employ the generalized algorithm with an intermediate p-value. For this purpose, the present chapter discusses the use of *iterative reweighted least squares* (IRLS) techniques to find the segmentation for any arbitrary p-norm ($1 < p < 3$) by solving a series of ℓ_2 optimizations. The use of IRLS hence allows one to find segmentation algorithms that are proper hybrids of existing segmentation algorithms such as graph cut, random walker, and shortest path.

8.1 A Generalized Image Segmentation Algorithm

A given image is represented by a weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The nodes $\mathcal{V}$ represent the pixels in the image and the edges $\mathcal{E}$ represent the choice of neighborhood structure. The weight of an edge, $e_{ij} \in \mathcal{E}$, is denoted by w_{ij}, and the weights are assumed here to be symmetric and nonnegative (i.e., $w_{ij} = w_{ji} \geq 0$).

Since this chapter assumes a scribble interface, it is assumed that some pixels in the image have been labeled as foreground and some others have been labeled as background. Let $\mathcal{M} \subset \mathcal{V}$ contain the locations of the nodes marked by the user and let $\mathcal{U} \subset \mathcal{V}$ contain the locations of the unmarked nodes. The set $\mathcal{M}$ is further divided into the sets $\mathcal{F} \subset \mathcal{M}$ and $\mathcal{B} \subset \mathcal{M}$ that contain the locations of the nodes labeled as the foreground object and the background, respectively. By construction, $\mathcal{M} \cap \mathcal{U} = \emptyset$, $\mathcal{M} \cup \mathcal{U} = \mathcal{V}$, $\mathcal{F} \cap \mathcal{B} = \emptyset$, and $\mathcal{F} \cup \mathcal{B} = \mathcal{M}$.

A Bernoulli random variable $y_i \in \{0, 1\}$ is defined for each node $i \in \mathcal{V}$, to indicate its binary segmentation as object ($y_i = 1$) or background ($y_i = 0$). A continuous-valued random variable x_i is introduced at each node to define the *success probability* for the distribution of the random variable y_i, that is, $p(y_i = 1 \,|\, x_i)$. For example, [143] uses logistic regression on a real-valued variable x_i to define the success probability as $p(y_i = 1 \,|\, x_i) = \frac{e^{x_i}}{1+e^{x_i}}$. In this chapter the success probability of y_i at node i is defined as

$$p(y_i = 1 \,|\, x_i) = \max\{\min\{x_i, 1\}, 0\} = \begin{cases} 1 & \text{if } x_i > 1, \\ x_i & \text{if } 0 \leq x_i \leq 1 \text{ and} \\ 0 & \text{if } x_i < 0. \end{cases} \tag{8.1}$$

For notational convenience, define vectors $\mathbf{x} \in \mathbb{R}^{|\mathcal{V}|}$ and $\mathbf{y} \in \mathbb{R}^{|\mathcal{V}|}$, whose ith entries are given by x_i and y_i, respectively. Now the goal is to infer the hidden variables, $\mathbf{x}$ and $\mathbf{y}$, from

the observed quantity (i.e., the image I). These hidden parameters can be estimated in a Bayesian framework by considering the following posterior probability model:

$$p(\mathbf{x}, \mathbf{y} \mid I) \propto p(\mathbf{x}) p(\mathbf{y} \mid \mathbf{x}) p(I \mid \mathbf{y}) = p(\mathbf{x}) \prod_{i \in \mathcal{V}} p(y_i \mid x_i) \prod_{i \in \mathcal{V}} p(I_i \mid y_i). \tag{8.2}$$

The term $p(\mathbf{x})$ is the prior term that encodes constraints on how the parameters of the Bernoulli variables vary spatially. Unlike most of the discussions in this book, in this chapter the smoothness constraints are enforced on the hidden variables $\mathbf{x}$ rather than on the segmentation $\mathbf{y}$ itself. The spatial smoothness prior is explicitly parameterized as

$$p(\mathbf{x}) \propto \exp\left(-\lambda \sum_{e_{ij} \in \mathcal{E}} (w_{ij} |x_i - x_j|)^p \right), \tag{8.3}$$

where $\lambda > 0$ and the weights w_{ij} are positive (i.e., $\forall e_{ij} \in \mathcal{E}$, $w_{ij} > 0$). Different choices for the p-norms result in different priors on $\mathbf{x}$. For example, $p = 1$ gives a Laplacian prior and $p = 2$ gives a Gaussian prior. The term $p(y_i \mid x_i)$ at each node is given completely by (8.1), where $p(y_i = 0 \mid x_i)$ is defined as $1 - p(y_i = 1 \mid x_i)$. The term $p(I_i \mid y_i)$ is the standard likelihood term as used in the rest of this book.

One of the drawbacks of the model discussed so far is that the edge weights for the pairwise terms, $\{w_{ij}\}$, do not depend on the image. Specifically, the edge weights are used as the parameters of the spatial prior model $p(\mathbf{x})$ and hence cannot depend on the image. However, as discussed in other chapters, in practice it is preferable to use contrast-sensitive edge weights, such as $w_{ij} = e^{-(I_i - I_j)^2}$, to align the segmentation boundary with the edges in the image. However, modifying the probabilistic model to accommodate contrast-sensitive weights is not straightforward. In the case of discrete MRFs, a modification of the likelihood term was proposed by Blake et al. [53], that better accommodates contrast-sensitive weights. However, it is unclear how such results would be applicable to the formulation of this chapter, which considers both continuous and discrete variables.

To this effect, this chapter follows an alternative formulation, which directly models the posterior, rather than attempting to decompose it in a likelihood and a prior term. Specifically, the posterior distribution of the hidden variables x and y is modeled as

$$\begin{aligned} p(\mathbf{x}, \mathbf{y}|I) &\propto p(\mathbf{y}|\mathbf{x}, I) p(\mathbf{x}|I) = p(\mathbf{y}|\mathbf{x}) p(\mathbf{x}|I) \\ &= \prod_{i \in \mathcal{V}} (x_i^{y_i} (1 - x_i)^{1 - y_i}) \prod_{i \in \mathcal{V}} \left(x_i^{-H(x_i - 0.5)} (1 - x_i)^{-(1 - H(x_i - 0.5))} \right) \\ &\quad \times \exp\left(-\lambda \sum_{e_{ij} \in \mathcal{E}} (w_{ij} |x_i - x_j|)^p \right) \exp\left(-\sum_{i \in \mathcal{V}} w_{i0}^p |x_i - 0|^p - \sum_{i \in \mathcal{V}} w_{i1}^p |x_i - 1|^p \right), \end{aligned} \tag{8.4}$$

where $H(\cdot)$ is the Heaviside function and $\forall i \in \mathcal{V}$, $w_{i0} \geq 0$ and $w_{i1} \geq 0$. The reduction of the term $p(\mathbf{y}|\mathbf{x}, I)$ to $p(\mathbf{y}|\mathbf{x})$ in the first line comes from the assumption that y is conditionally independent of I, given x. The terms introduced in the bottom row of (8.4) act as the unary terms and the weights w_{i0} and w_{i1} bias the parameters x_i towards 0 and 1. Firstly, these unary terms serve the purpose of encoding the user's interaction. If a node $i \in \mathcal{M}$ is labelled as object or backgraound, the algorithm sets the corresponding unary terms as $(w_{i0}, w_{i1}) = (0, \infty)$ or $(w_{i0}, w_{i1}) = (\infty, 0)$, respectively. It can be verified that this is equivalent to hardcoding the value of x_i at the marked nodes $i \in \mathcal{M}$ as $\forall i \in \mathcal{F}$, $x_i = 1$ and $\forall i \in \mathcal{B}$, $x_i = 0$. The unary terms may also be used to encode the extent to which the appearance (color, texture, etc.) of a node i obeys an appearance model for the object or the backgraound.

Given the expression (8.4), the goal is now to estimate the hidden variables $\mathbf{x}$ and $\mathbf{y}$ as $\underset{\mathbf{x},\mathbf{y}}{\text{argmax}}\, p(\mathbf{x}, \mathbf{y}|I)$. It can be verified that estimating $\hat{\mathbf{y}} = \underset{\mathbf{y}}{\text{argmax}}\, p(\mathbf{x}, \mathbf{y}|I)$ gives for each node $i \in \mathcal{V}$, $y_i = 1$ if $x_i \geq 0.5$ and $y = 0$ otherwise. It can also be verified that estimating the optimal value of $\hat{\mathbf{x}}$ as $\underset{\mathbf{x}}{\text{argmax}}\, p(\mathbf{x}, \hat{\mathbf{y}}|I)$, is equivalent to estimating $\hat{\mathbf{x}}$ as the minimizer of the energy function $E(\mathbf{x})$, where $E(\mathbf{x})$ is defined as

$$E(\mathbf{x}) = \sum_{i \in \mathcal{V}} w_{i0}^p |x_i - 0|^p + \sum_{i \in \mathcal{V}} w_{i1}^p |x_i - 1|^p + \lambda \sum_{e_{ij} \in \mathcal{E}} (w_{ij}|x_i - x_j|)^p, \tag{8.5}$$

where $\lambda > 0$ is the same parameter as in (8.4), that accounts for the trade-off between the unary terms and the pairwise terms.

For notational convenience, one can introduce two auxiliary nodes for the foreground and the background: f and b, respectively. The parameters x_i at these nodes are hardcoded as $x_f = 1$ and $x_b = 0$. The unary terms can then be rewritten as $w_{i0}^p |x_i - 0|^p = w_{i0}^p |x_i - x_b|^p$ and $w_{i1}^p |x_i - 1|^p = w_{i1}^p |x_i - x_f|^p$. Hence, without loss of generality, $E(\mathbf{x})$ can be rewritten in terms of pairwise interactions only, as

$$E(\mathbf{x}) = \sum_{e_{ij} \in \mathcal{E}} (w_{ij}|x_i - x_j|)^p, \tag{8.6}$$

where, with abuse of notation, the set $\mathcal{E}$ is modified to include the original set of edges $\mathcal{E}$ defined in (8.5), as well as the additional edges introduced by representing the unary terms as pairwise interactions.

Now, note that $E(\mathbf{x})$ is parameterized by a finite-valued p-norm, $p < \infty$. The limiting case $p = \infty$ is admitted by generalizing $E(\mathbf{x})$ as

$$E_p(\mathbf{x}) = \left[\sum_{e_{ij} \in \mathcal{E}} (w_{ij}|x_i - x_j|)^p \right]^{\frac{1}{p}}. \tag{8.7}$$

Due to the monotonicity of the $(\cdot)^{\frac{1}{p}}$ operator, (8.6) and (8.7) have the same optimum for a finite $0 < p < \infty$. As shown later, the generalization to $p = \infty$ allows the shortest path segmentation algorithm to be admitted as a special case of the generalized algorithm for $p = \infty$.

Therefore, the problem of computing the segmentation is recast as the problem of computing the optimal $\hat{\mathbf{x}}$ that minimizes $E_p(\mathbf{x})$, subject to the constraints enforced by the user's interaction, as

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} E_p(\mathbf{x}) \text{ s.t. } x_i = 1, \text{ if } i \in \mathcal{F} \text{ and } x_i = 0, \text{ if } i \in \mathcal{B}. \tag{8.8}$$

It is shown later that the solution to (8.8) naturally satisfies the constraint that $\forall i \in \mathcal{U}, 0 \leq x_i \leq 1$. We can redefine $\mathbf{y}$ to be the *hard* segmentation produced from the real-valued $\mathbf{x}$ by thresholding at $x = 0.5$. This is equivalent to obtaining the segmentation of node i as $\hat{y}_i = \arg\max_{y_i} p(y_i \mid \hat{x}_i)$. This segmentation procedure is summarized in algorithm 8.1. This generalized segmentation algorithm is referred to as the **p-brush** algorithm, due to the dependence of the solution on the p-norm. It will be shown later that the graph cut, random walker, and shortest path segmentation algorithms can be viewed as special instances of this p-brush algorithm when $p = 1, 2$, and ∞, respectively.

Algorithm 8.1 *(p-Brush: A Generalized Image Segmentation Algorithm)*

Given:

- Two sets of pixels marked for the foreground object ($\mathcal{F} \in \mathcal{V}$) and the background ($\mathcal{B} \in \mathcal{V}$).
- A norm $p \geq 1$ for the energy function $E_p = \left[\sum_{e_{ij} \in \mathcal{E}} (w_{ij}|x_i - x_j|)^p\right]^{\frac{1}{p}}$ that includes unary terms as well as the spatial prior.

Compute: $\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} E_p(\mathbf{x})$, s.t. $x_i = 1$ if $i \in \mathcal{F}$ and $x_i = 0$ if $i \in \mathcal{B}$.

Output: Segmentation $\mathbf{y}$ defined as $\hat{y}_i = 1$ if $\hat{x}_i \geq \frac{1}{2}$ and $\hat{y}_i = 0$ if $\hat{x}_i < \frac{1}{2}$.

8.2 Solutions to the p-Brush Problem

This section initially discusses some interesting properties of the solutions of the p-brush algorithm. The remaining discussion focuses on efficient solvers for particular values of the p-norm.

An important property of $E_p(\mathbf{x})$ is its convexity for all values of $p \geq 1$. Therefore, any solution of (8.8) must be a global minimizer of $E_p(\mathbf{x})$. In what follows, a global minimizer of $E_p(\mathbf{x})$ is denoted as $\hat{\mathbf{x}}_p$ and its ith entry is denoted as $\hat{x}_{p,i}$. It was shown in [440] that these minimizers have interesting properties.

Extremum Value Property The value of $\hat{x}_{p,i}$ at every node $i \in \mathcal{V}$ is bounded by the values of $\hat{x}_{p,j}$ at the marked nodes $j \in \mathcal{M}$. Formally, this can be written as $\forall i \in \mathcal{V}$, $\min_{j \in \mathcal{M}} \hat{x}_{p,j} \leq \hat{x}_{p,i} \leq \max_{j \in \mathcal{M}} \hat{x}_{p,j}$.

Now recall that by construction, the entries of $\hat{\mathbf{x}}_p$ are fixed at the marked nodes as $\forall i \in \mathcal{F}$, $\hat{x}_{p,i} = 1$, and $\forall i \in \mathcal{B}$, $\hat{x}_{p,i} = 0$. Hence, the extremum value property can be used to conclude that the value of $\hat{x}_{p,i}$ at each unmarked node $i \in \mathcal{U}$ lies in $[0, 1]$. As a result, the set of solutions for the entries of $\hat{\mathbf{x}}_p$ at the unmarked nodes $\mathcal{U}$ is $[0, 1]^{|\mathcal{U}|}$, which is a compact and convex set. This result, coupled with the fact that the energy function $E_p(\mathbf{x})$ is convex in $\mathbf{x}$, implies that any descent algorithm can be used to calculate the global minimizer of $E_p(\mathbf{x})$.

Right Continuity in the p-Norm This property characterizes the continuity of the solutions of the p-brush algorithm as a function of the p-norm. In particular it can be shown that if $\hat{\mathbf{x}}_{p+\epsilon}$ is a minimizer of $E_{p+\epsilon}(\mathbf{x})$, where $\epsilon \geq 0$, then $\hat{\mathbf{x}}_{p+\epsilon}$ is right continuous in p, that is, $\lim_{\epsilon \to 0^+} \hat{\mathbf{x}}_{p+\epsilon} = \hat{\mathbf{x}}_p$. The significance of this property will be illustrated later while discussing the IRLS algorithm for estimating the solutions of the p-brush algorithm for the range $1 < p < 3$.

8.2.1 Special Cases of the p-Brush Algorithm

Before studying the case for a general p, it is of interest to study the instances of the p-brush algorithm resulting from the choices of $p = 1, 2$, and ∞ since they correspond to existing segmentation algorithms.

The $p = 1$ Case: Graph Cut After substituting $p = 1$ in the p-brush algorithm, the second step of the algorithm requires the solution to the problem

$$\min_{\mathbf{x}} \sum_{e_{ij} \in \mathcal{E}} w_{ij} |x_i - x_j|, \text{ s.t. } x_i = 1 \text{ if } i \in \mathcal{F}, \text{ and } x_i = 0 \text{ if } i \in \mathcal{B}. \tag{8.9}$$

It is known that the problem in (8.9) admits a purely binary solution, $x_i \in \{0, 1\}$, due to the totally unimodular property of the min-cut problem [363]. This is precisely the solution that is produced by the graph cut algorithm using the min-cut/max-flow solver [66]. Notice that (8.9) provides a continuous-valued interpretation of the graph cut algorithm as opposed to the traditional discrete-valued interpretation.

Although (8.9) admits a purely binary solution, the solution may not be unique and there may be continuous-valued nonbinary solutions to (8.9). A result in [85] can be used to obtain a purely binary-valued minimizer from any continuous-valued minimizer. Specifically, a binary-valued minimizer ($\hat{\mathbf{x}}_1^B \in \{0, 1\}^{|\mathcal{V}|}$) can be produced from a continuous-valued minimizer ($\hat{\mathbf{x}}_1^C \in [0, 1]^{|\mathcal{V}|}$) of (8.9) by thresholding its entries at any value $\nu \in (0, 1)$, that is, $\forall i \in \mathcal{V}$: $\hat{x}_{1,i}^B = 1$ if $\hat{x}_{1,i}^C \geq \nu$, and $\hat{x}_{1,i}^B = 0$ otherwise. It was shown in [85] that both solutions, $\hat{\mathbf{x}}_1^B$ and $\hat{\mathbf{x}}_1^C$, are minimizers of (8.9). Hence, thresholding any solution to (8.9)

at $\nu = 0.5$ produces a valid minimum cut. It is interesting that although this model deals with continuous-valued solutions as opposed to discrete MRF models (e.g., chapter 7), the thresholded solution is indeed equivalent to the solution of the discrete model.

The $p = 2$ Case: Random Walker For the case $p = 2$, the second step of the p-brush algorithm requires the solution to the problem

$$\min_{\mathbf{x}} \sum_{e_{ij} \in \mathcal{E}} w_{ij}^2 \left(x_i - x_j\right)^2, \text{ s.t. } x_i = 1 \text{ if } i \in \mathcal{F}, \text{ and } x_i = 0 \text{ if } i \in \mathcal{B}. \tag{8.10}$$

This is exactly the optimization problem solved by the random walker algorithm in [175] (for the case of two labels). A random walk is defined on the graph such that the probability that a random walker at node $i \in \mathcal{V}$ moves to a neighboring node $j \in \mathcal{N}_i$ is given as $w_{ij} / \sum_{k \in \mathcal{N}_i} w_{ik}$. The random walk is terminated when the random walker reaches any of the marked nodes. [175] showed that the solution of (8.10), $\hat{\mathbf{x}}_2$, satisfies the property that $\hat{x}_{2,i}$ corresponds to the probability that a random walker starting from node $i \in \mathcal{V}$ will reach a node labeled as foreground before a node labeled as background. These probabilities are thresholded at $x = 0.5$ to obtain the segmentation. Hence, the random walker algorithm with two labels gives the same solution as the p-brush algorithm when $p = 2$.

The $p = \infty$ Case: Shortest Path When $p \to \infty$, the limit of $E_p(\mathbf{x})$ is given as

$$\lim_{p \to \infty} E_p(\mathbf{x}) = \underbrace{\max_{e_{ij} \in \mathcal{E}} w_{ij} |x_i - x_j|}_{\rho(\mathbf{x})} \underbrace{\lim_{p \to \infty} \sqrt[p]{\sum_{e_{ij} \in \mathcal{E}} \left(\frac{w_{ij} |x_i - x_j|}{\rho(\mathbf{x})} \right)^p}}_{1} = \rho(\mathbf{x}), \tag{8.11}$$

where $\rho(\mathbf{x})$ is defined as $\rho(\mathbf{x}) = \max_{e_{ij} \in \mathcal{E}} w_{ij} |x_i - x_j|$. Now the optimization problem in (8.11) can be rewritten as

$$\min_{\mathbf{x}} \left[\max_{e_{ij} \in \mathcal{E}} (w_{ij} |x_i - x_j|) \right], \text{ s.t. } x_i = 1 \text{ if } i \in \mathcal{F}, \text{ and } x_i = 0 \text{ if } i \in \mathcal{B}. \tag{8.12}$$

This optimization problem may be viewed as a combinatorial formulation of the minimal Lipschitz extension problem [15]. It has been shown that the solution to (8.12) is not unique in general [15]. Theorem 8.1 provides one possible interesting construction to minimize $E_\infty(\mathbf{x})$.

Theorem 8.1 **Infinity-Norm Optimization** *Define the distance between neighboring nodes i and j as $d_{ij} = \frac{1}{w_{ij}}$. Denote the shortest path lengths from node $i \in \mathcal{V}$ to a node marked foreground and background as d_i^F and d_i^B, respectively. The vector, $\hat{\mathbf{x}}_\infty$, defined as $\forall i \in \mathcal{U} : \hat{x}_{\infty,i} = d_i^B / d_i^B + d_i^F$, is a solution to* (8.12).

Proof Given in the appendix to this chapter. ∎

Note that a node $i \in \mathcal{V}$ is assigned to the foreground if $\hat{x}_{\infty,i} > 0.5$ (i.e., $d_i^F < d_i^B$). This implies that the segmentation given by the shortest path algorithm [18] is a valid solution to the p-brush algorithm for the case $p = \infty$. Hence $\hat{\mathbf{x}}_\infty$ may be computed efficiently using Dijkstra's shortest path algorithm. However, as mentioned earlier, this is not the only solution to (8.12). One could introduce other constructions and additional regularizers, as in [441], to obtain a unique solution.

8.2.2 Segmentation with an Arbitrary p-Norm

In the special cases discussed so far, a specific solver is used for each case due to the properties of the employed p-norm. For any arbitrary finite $p \in (1, 3)$, algorithm 8.2 can be used to estimate $\hat{\mathbf{x}}_p$, employing iterative reweighted least squares (IRLS). Each iteration of the algorithm has two steps. The first step, *reweighting*, involves the update of the weights based on the current estimate of $\mathbf{x}$ (see (8.13)). The second step, *least squares estimation*, involves updating the value of $\mathbf{x}$ by solving a least squares problem with the updated weights (see (8.14)). IRLS can also be thought of as an iterative random walker algorithm with the weights being updated at each iteration. The rationale behind the algorithm is as follows. For $p > 1$, the function $(E_p(\mathbf{x}))^p$ is differentiable. In this case, algorithm 8.2 is equivalent to performing a Newton descent of $(E_p(\mathbf{x}))^p$ with step size $(p-1)$ [440]. This is because, when $\forall e_{ij} \in \mathcal{E}$, $x_i^{(n)} \neq x_j^{(n)}$, the matrix $W^{(n)}$, whose (i, j) entry is given by $w_{ij}^{(n)}$, is exactly the Hessian (say $H_p(\mathbf{x})$) of $(E_p(\mathbf{x}))^p$ evaluated at $\mathbf{x} = \mathbf{x}^{(n)}$. If $x_i = x_j$ for some $e_{ij} \in \mathcal{E}$, then

---

Algorithm 8.2 *(Estimation of $\hat{\mathbf{x}}_p$ for Any $1 < p < 3$, Using IRLS)*

---

1. Set $n = 0$ and choose a value $\alpha > 0$ and a stopping criterion $\delta > 0$.
2. Initialize the membership vector $\mathbf{x}^{(0)}$ as $\forall i \in \mathcal{F} : x_i^{(0)} = 1$, $\forall i \in \mathcal{B} : x_i^{(0)} = 0$, and $\forall i \in \mathcal{U} : x_i^{(0)} = 0.5$.
3. For each edge $\mathbf{e}_{ij} \in \mathcal{E}$, define the edge weight as $w_{ij}^{(n)}$:

$$w_{ij}^{(n)} = \begin{cases} w_{ij}^p |x_i^{(n)} - x_j^{(n)}|^{p-2} & \text{if } x_i \neq x_j \\ \alpha^{p-2} & \text{if } x_i = x_j. \end{cases} \tag{8.13}$$

4. Calculate $\mathbf{x}^{(n+1)}$ as the solution of

$$\arg\min_{\mathbf{x}} \sum_{e_{ij} \in E} w_{ij}^{(n)} \left(x_i - x_j\right)^2, \qquad \text{s.t. } x_i = 0/1 \text{ if } i \in \mathcal{F}/\mathcal{B}. \tag{8.14}$$

5. If $|\mathbf{x}_U^{(n+1)} - \mathbf{x}_U^{(n)}| > \delta$, update $n = n + 1$ and go to step 3.

$H_p(\mathbf{x})$ does not exist for $1 < p < 2$. This is resolved by approximating $H_p(\mathbf{x})$, using the weights defined in (8.13). It can be verified that IRLS still produces a descent direction for updating $\mathbf{x}^{(n)}$ at each step.

For $p = 1$, $E_1(\mathbf{x})$ is not differentiable. However, recall from the properties of $\hat{\mathbf{x}}_p$ discussed earlier in this section that the minimizers of $E_p(\mathbf{x})$ are right continuous with respect to the p-norm. Therefore, the minimizer of $(E_{1+\epsilon}(\mathbf{x}))^{1+\epsilon}$ can be calculated using IRLS and used as an approximation of $\hat{\mathbf{x}}_1$ with a desired accuracy by choosing ϵ to be sufficiently small. In general, IRLS is provably convergent only for $1 < p < 3$ [359]. However, solutions for $p \geq 3$ can be obtained by using Newton descent with an adaptive step size rather than $p - 1$.

8.3 Experiments

This section evaluates the performance of the p-brush algorithm on synthetic medical as well as natural images. The experiments aim to highlight common problems in the segmentation results and to analyze how they might depend on the choice of the p-norm. The results first present the image to be segmented and then show the segmentation results obtained for various values of p. The segmentation boundary is superimposed on the image either as a dashed line or as a bold line to ensure visibility. The user interaction for the object and background is superimposed in different shades of gray to distinguish them.

A practical segmentation system has many components that can significantly affect its performance, such as unary terms and neighborhood. Most existing algorithms rely on good unary terms that, on their own, produce near-perfect segmentations. When the unary terms are uninformative, the segmentation relies primarily on the spatial prior. Therefore, unary terms are ignored in the following evaluation to isolate and analyze the effect of the p-norm on the spatial regularization of the segmentation boundary. It should, however, be noted that unary terms have been employed in existing literature for $p = 1$ [67], $p = 2$ [174], and $p = \infty$ [18]. Though it is not within the goals of this chapter, it is of future interest to analyze how the unary terms behave for general values of p. Along the same lines, a simple 4-connected lattice is used to define the neighborhood in order to inspect metrication artifacts that might otherwise be eliminated by considering higher connected neighborhoods. The contrast-sensitive weight for an edge $e_{ij} \in \mathcal{E}$ is defined as $w_{ij} = e^{-\beta \|I_i - I_j\|}$, where $\beta > 0$ and I_i is the gray scale intensity or RGB color of pixel i.

8.3.1 Metrication Artifacts

The artifacts correspond to blocky segmentations that follow the topology of the neighborhood structure. They occur in the case of $p = 1$ and $p = \infty$, due to the neighborhood structure [67]. In contrast, the case $p = 2$ corresponds to a finite differences discretization of a continuous (inhomogeneous) Laplace equation. Chapter 12 shows that discretization of continuous formulations can reduce metrication errors. Hence, it may be conjectured that metrication artifacts are reduced in the case $p = 2$.

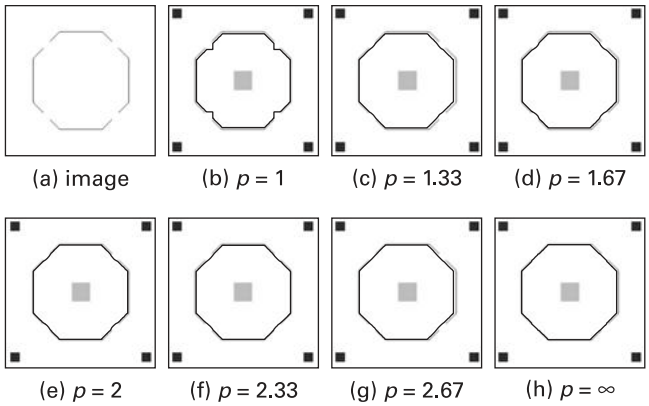

Figure 8.1
Analysis of metrication artifacts in the segmentation of the synthetic image shown in (a). Light and dark marks (squares in this example) indicate user labelings of foreground and background, respectively. The segmentations obtained for various p-norms are shown in (b)–(h). Blocky artifacts are present for $p = 1$ but not for higher values of p.

To understand this better, consider the experiments in figure 8.1, where the goal is to segment the octagon in figure 8.1a. Graph cut ($p = 1$) produces a squared-off effect at the portion where the octagon's boundary has been erased. This is the metrication artifact. The artifacts are absent as p increases from 1 to 2.67. Although they are avoided for $p = \infty$ in this synthetic example, they are exhibited for $p = \infty$ in the examples on real images. Figure 8.2 shows metrication artifacts in the segmentation of an aneurysm in an MR image. The metrication artifacts are clearly visible for $p = 1$. They seem to reduce as p increases, but not as drastically as in the previous example. Specifically, for $p = 1.25$, the segmentation boundary to the left of the aneurysm is less blocky compared with the result for $p = 1$. However, the boundary is still blocky at the top and bottom of the aneurysm. These artifacts reduce as p increases from 1 to 2.75, but they reappear for $p = \infty$. The same trend can be seen in the results in figure 8.5.

In general, the metrication artifacts may be reduced by choosing a higher connected neighborhood [67]. For example, a 8-neighborhood instead of a 4-neighborhood would have given the desired result for $p = 1$ in the example in figure 8.1. However, increasing the connectivity is equivalent to increasing the number of edges in the graph, and hence comes at the expense of higher memory usage and higher computation.

8.3.2 Proximity Bias

This bias corresponds to the sensitivity of the segmentation to the location of the user's interaction. The proximity bias is best understood in $p = \infty$, since the segmentation of an unmarked pixel depends on its distance from the marked pixels. It was shown in [175]

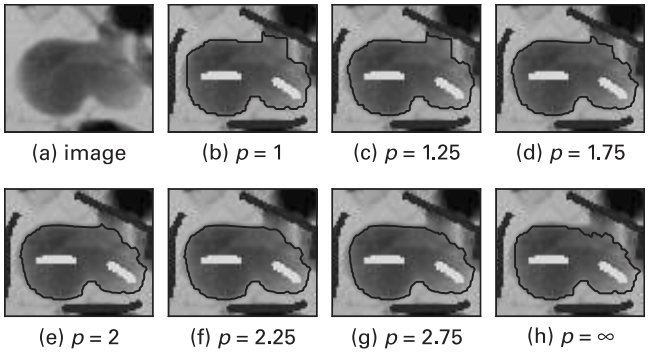

Figure 8.2
Analysis of metrication artifacts in the segmentation of the medical image shown in (a). The segmentations obtained for various p-norms are shown in (b)–(h). The artifacts reduce as p increases from 1 to 2.75, but are present for $p = \infty$.

that for $p = 2$, the segmentation of an unmarked pixel depends on the distances of all the parallel paths from that pixel to the marked pixels, thus reducing dependence on a single path. No such interpretation is known for $p = 1$.

To understand this, consider the experiments in figure 8.3, where the goal is to segment the image into equal halves along the vertical dark line. The user's scribbles are not symmetric with respect to the line. If the scribbles were symmetric, the segmentation computed for the various p-norms would be as desired. In this case the desired segmentation is obtained for $p = 1$ and $p = 1.33$. As p increases, the segmentation begins to *leak* through the portions where the dark line has been erased. This is the proximity bias.

This bias is further explored in the results in figure 8.4. In contrast to the user interaction in figure 8.2, the one for the aneurysm has been erased toward the bottom right. This is done to analyze the effect on the segmentation boundary near the weak edge at the bottom right of the aneurysm. It can be seen that for $p = 1$ and $p = 1.1$, the segmentation boundary is correctly aligned with the aneurysm. However, as p increases, the segmentation begins to leak and the boundary is incorrectly aligned with the aneurysm's interior.

In general, the proximity bias can be reduced by further user interaction to correct any errors. However, this might require greater levels of interaction from the user, which can prove to be a burden for real-time applications. Moreover, additional user interaction might not be possible in unsupervised applications, where the user interaction is automatically generated by the algorithm, based on appearance models learned a priori for the object and background.

8.3.3 Shrinking Bias

The shrinking bias corresponds to the bias toward segmentation boundaries with shorter length. This can be understood better for $p = 1$ (i.e., graph cut), because it has been shown

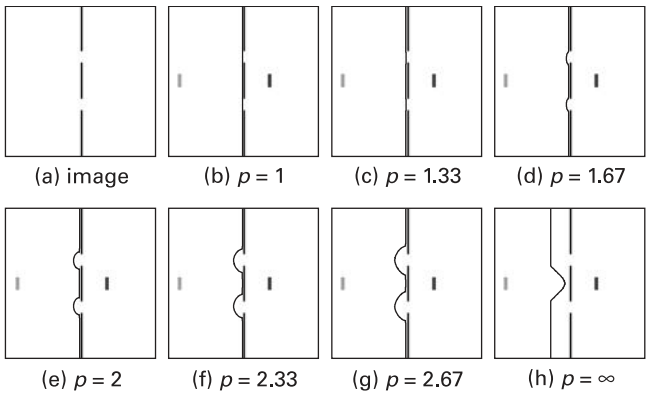

Figure 8.3
Analysis of proximity bias in the segmentation of the synthetic image shown in (a). The segmentations obtained for various p-norms are shown in (b)–(h). The desired segmentation is produced for $p = 1$ and $p = 1.33$. As p increases, the segmentation boundary leaks through the erased portions of the vertical line and gradually moves toward the distance based segmentation produced when $p = \infty$.

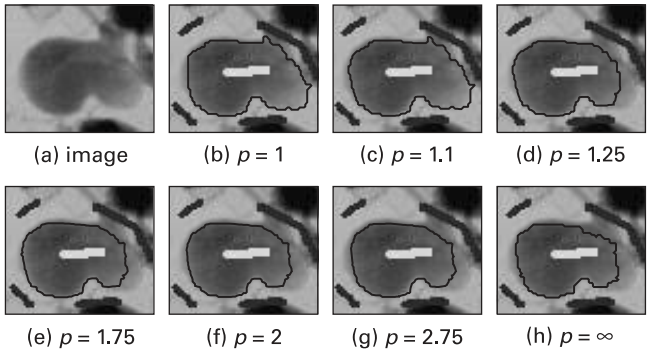

Figure 8.4
Analysis of proximity bias in the segmentation of the image shown in (a). The segmentation boundary does not leak for $p = 1$ and $p = 1.1$. However, as p increases, the proximity bias increases and the segmentation boundary leaks.

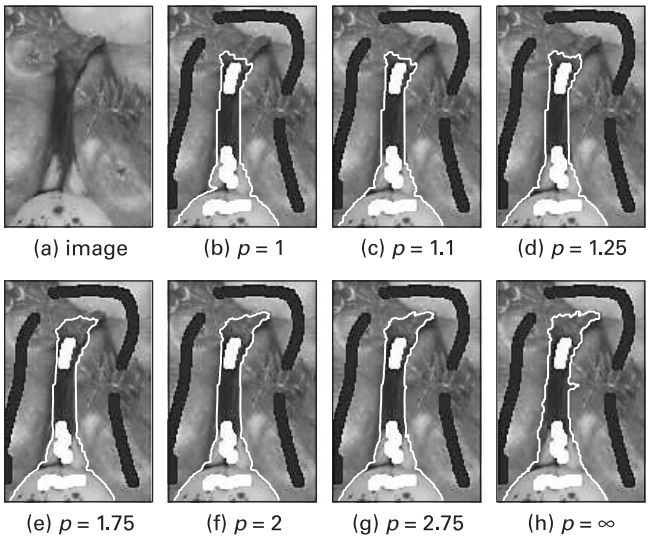

Figure 8.5
Analysis of shrinking bias in the segmentation of a banana in the natural image shown in (a). The segmentations obtained for various p-norms are shown in (b)–(h). The tip of the banana is cut off for $p = 1, 1.1, 1.25$, and 1.75, with the portion being cut off gradually reducing. As p increases, the shrinking bias seems to reduce and the segmentation boundary aligns with the banana. Also, the metrication artifacts along the right boundary decrease as p increases.

that the smoothness prior, that is, $p(\mathbf{x})$ in (8.3), can be viewed as a penalty on the length of the segmentation boundary [67]. For higher values of p, there is no such known relationship between the boundary length and the prior. Due to this bias, the segmentation boundary might collapse, thereby resulting in a *shortcutting* of the segmentation boundary.

In order to appreciate this, consider the results in figure 8.5, where the goal is to segment the top portion of a banana. The top of the banana is cut off for $p = 1, 1.1, 1.25$, and 1.75, with the portion being cut off gradually reducing. Also, the segmentation boundary toward the central left portion of the banana incorrectly aligns with the edges inside the banana. As p increases, the segmentation begins to align correctly with the boundary of the banana. The shrinking bias reduces as p increases.

This trend can also be seen in figure 8.6, where the goal is to segment the ultrasound image shown in figure 8.6a along the interface of the dark and bright portions of the image. The shortcutting effect (i.e., the shrinking bias) reduces as p increases.

In general, such errors caused by shortcutting of the segmentation boundary can be resolved by further user interaction. However, as mentioned earlier, higher levels of user interaction are not preferable for real-time segmentation or unsupervised segmentation.

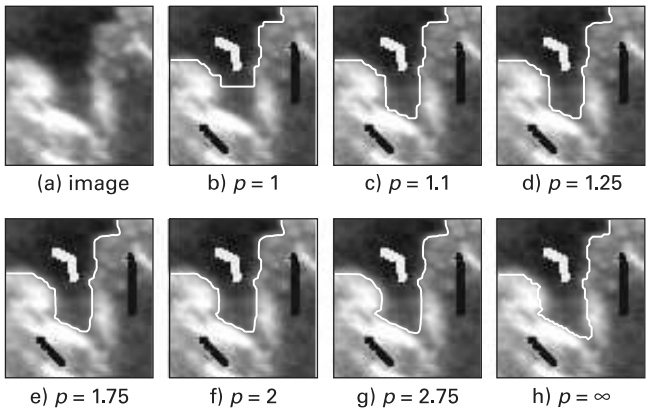

Figure 8.6
Analysis of shrinking bias in the segmentation of the medical image shown in (a). The goal is to segment the interface between the bright and the dark regions. The segmentations obtained for various p-norms are shown in (b)–(h). As p increases, the shrinking bias seems to reduce.

8.4 Conclusion

The p-brush segmentation algorithm provides a generalized framework that includes existing algorithms such as graph cut, random walker, and shortest path as special cases for particular values of $p = 1, 2$, and ∞, respectively. Due to the nature of their cost functions, these algorithms have specific efficient solvers and characteristic drawbacks.

The experiments suggest that there is a correlation between the discussed biases and the p-norm. Specifically, the proximity bias increases and the shrinking bias decreases as p increases. Metrication artifacts are observed for $p = 1$ and $p = \infty$, but not for $p = 2$. Since $\hat{\mathbf{x}}_p$ is continuous in p, it is conjectured that these artifacts reduce as p increases from 1 to 2, but reappear for p beyond some $\tilde{p} > 2$. Due to the interplay of these issues, it cannot be determined beforehand which p-norm would be the best for a general segmentation system. However, if the system is to be used for a specific segmentation goal, an optimal value for p may be learned through a user study or by training on representative exemplar images. IRLS may be used to obtain the segmentation for several values of p with the aim of obtaining a trade-off among the properties of these three special cases. However, IRLS may be potentially slower than the efficient solvers for the cases of $p = 1, 2$, and ∞.

8.5 Appendix: Proof of Infinity Norm Optimization Theorem

In order to verify that $\hat{\mathbf{x}}_\infty$ defined in the hypothesis is indeed a solution of (8.12), it must satisfy the following two conditions.

1. $\hat{\mathbf{x}}_{\infty,i} = 1$ or $\hat{\mathbf{x}}_{\infty,i} = 0$ if the pixel i is labeled as foreground ($i \in \mathcal{F}$) or as background ($i \in \mathcal{B}$), respectively.

This condition can be verified very easily. Note that $d_i^F = 0$ and $d_i^B > 0$ for all the pixels i labeled as belonging to the foreground. This implies that $\forall i \in \mathcal{F} : \hat{\mathbf{x}}_{\infty,i} = \frac{d_i^B}{d_i^B + 0} = 1$. Since $d_i^B = 0$ and $d_i^F > 0$ for all the pixels i labeled as belonging to the background, this implies that $\forall i \in \mathcal{B} : \hat{\mathbf{x}}_{\infty,i} = \frac{0}{0 + d_i^F} = 0$.

2. If $\mathbf{x}_\infty^*$ is a solution to (8.12), then $\rho(\hat{\mathbf{x}}_\infty) = \rho(\mathbf{x}_\infty^*)$.

In order to verify this, it shall be proved that $\rho(\hat{\mathbf{x}}_\infty) \leq \rho(\mathbf{x}_\infty^*)$. The definition of $\mathbf{x}_\infty^*$ being a solution to (8.12) would imply that $\rho(\hat{\mathbf{x}}_\infty) = \rho(\mathbf{x}_\infty^*)$, and the proof would be complete. Since $\rho(\hat{\mathbf{x}}_\infty)$ is defined as $\max_{\mathbf{e}_{ij} \in \mathcal{E}}(w_{ij}|\hat{\mathbf{x}}_{\infty,i} - \hat{\mathbf{x}}_{\infty,j}|)$, it is sufficient to show that $\forall \mathbf{e}_{ij} \in \mathcal{E} : w_{ij}|\hat{\mathbf{x}}_{\infty,i} - \hat{\mathbf{x}}_{\infty,j}| \leq \rho(\mathbf{x}_\infty^*)$ to prove that $\rho(\hat{\mathbf{x}}_\infty) \leq \rho(\mathbf{x}_\infty^*)$.

Now, for any edge $\mathbf{e}_{ij} \in \mathcal{E}$, one can conclude from the triangle inequality that $|d_i^B - d_j^B| \leq w_{ij}^{-1}$ and $|d_i^F - d_j^F| \leq w_{ij}^{-1}$. This can be used to derive the following inequalities.

$$
\begin{aligned}
& w_{ij}|\hat{\mathbf{x}}_{\infty,i} - \hat{\mathbf{x}}_{\infty,j}| \\
&\quad = w_{ij}\frac{|d_i^B(d_j^F - d_i^F) + d_i^F(d_i^B - d_j^B)|}{(d_j^B + d_j^F)(d_i^F + d_i^B)} \text{ (rearranging the terms)} \\
&\quad \leq w_{ij}\frac{d_i^B|d_j^F - d_i^F| + d_i^F|d_i^B - d_j^B|}{(d_j^B + d_j^F)(d_i^F + d_i^B)} \text{ (using triangle inequality)} \\
&\quad \leq \frac{d_i^B + d_i^F}{(d_j^B + d_j^F)(d_i^F + d_i^B)} \quad \left(\text{since } |d_i^B - d_j^B| \leq w_{ij}^{-1} \text{ and } |d_i^F - d_j^F| \leq w_{ij}^{-1}\right) \\
&\quad = \frac{1}{d_j^B + d_j^F}.
\end{aligned}
\tag{8.15}
$$

In order to complete the proof, a result from [441] will be used to show that $\forall k \in \mathcal{V}$, $\frac{1}{d_k^B + d_k^F} \leq \rho(\mathbf{x}_\infty^*)$. Specifically, let $\pi : u \rightsquigarrow v$ denote a path π in $\mathcal{G}$ from node $u \in \mathcal{V}$ to $v \in \mathcal{V}$. It was shown in [441] that

$$
\rho(\mathbf{x}_\infty^*) \geq \left(\sum_{e_{ij} \in \pi} w_{ij}^{-1}\right)^{-1}, \forall \pi : f \rightsquigarrow b, \text{ where } f \in \mathcal{F} \text{ and } b \in \mathcal{B}. \tag{8.16}
$$

Now, consider any node $k \in \mathcal{V}$ and denote the marked nodes labeled as foreground and background that are closest to this node k as $f_k \in \mathcal{F}$ and $b_k \in \mathcal{B}$, respectively. Denote the shortest path from f_k to k as $\pi_{f_k,k} : f_k \rightsquigarrow k$ and the shortest path from k to b_k as $\pi_{k,b_k} : k \rightsquigarrow b_k$. Now, consider the path $\pi_{f_k,b_k} : f_k \rightsquigarrow b_k$ from f_k to b_k that is obtained by traversing from f_k to k along $\pi_{f_k,k}$ and then from k to b_k along π_{k,b_k}. By using (8.16), it can

be seen that $\rho(\mathbf{x}^*_\infty) \geq \left(\sum_{e_{ij}\in\pi_{f_k,b_k}} w_{ij}^{-1}\right)^{-1} = (d_k^F + d_k^B)^{-1}$. Since this holds true for every node $k \in \mathcal{V}$, it can be seen that

$$\forall k \in \mathcal{V}, \frac{1}{d_k^B + d_k^F} \leq \rho(\mathbf{x}^*_\infty). \tag{8.17}$$

Hence it can be concluded from (8.15) and (8.17) that $\forall \mathbf{e}_{ij} \in \mathcal{E} : w_{ij}|\hat{\mathbf{x}}_{\infty,i} - \hat{\mathbf{x}}_{\infty,j}| \leq \rho(\mathbf{x}^*_\infty)$. The proof is complete with this result.

Acknowledgments

The authors Dheeraj Singaraju and Rene Vidal would like to thank the Office of Naval Research, USA for supporting this work through the grant ONR YIP N00014-09-1-0839. The authors would also like to thank Donald Geman for his helpful discussions about the probability model for the p-brush algorithm.