A New Probabilistic Relaxation Scheme and Its Application to Edge Detection

Weian Deng and S. Sitharama Iyengar

Abstract—This paper presents a new scheme for probabilistic relaxation labeling that consists of an update function and a dictionary construction method. The nonlinear update function is derived from Markov Random Field theory and Bayes' formula. The method combines evidence from neighboring label assignments and eliminates label ambiguity efficiently. This result is important for a variety of image processing tasks, such as image restoration, edge enhancement, edge detection, pixel classification, and image segmentation.

We successfully applied this method to edge detection. The relaxation step of the proposed edge-detection algorithm greatly reduces noise effects, gets better edge localization such as line ends and corners, and plays a crucial role in refining edge outputs. The experiments show that our algorithm converges quickly and is robust in noisy environments.

Index Terms—Probabilistic relaxation, dictionary scheme, MRF, edge detection.

1 Introduction

THE past decade has seen an explosive growth in the study of relaxation labeling techniques for image processing. The integration of contextual information with conventional image processing techniques is an important research topic. Relaxation labeling uses contextual information to resolve object labeling ambiguities as locally as possible. The amount of contextual information employed is expanded recursively until a unique labeling results. Relaxation labeling has been applied to a variety of image processing problems, such as image restoration [3], edge enhancement [17], edge detection [5], [6], [7], [16], pixel classification [2], and image segmentation [8].

Given a set of objects(or vertices) $V = \{1, 2, ..., n\}$, a graph $G = (V, E)$ that describes neighborhood relations among objects in V, a set of labels $\Lambda = \{\lambda_1, \lambda_2, ..., \lambda_m\}$, a set of noisy observations $Y = \{\bar{y}_i = h(x_i, u_i) \mid i \in V\}$ ($\bar{y}_i$ is the vector of observations for object i that depends on the true label assignment x_i and the random noise u_i. x_i and u_i are assumed to be independent and noise degradation is assumed to be object-independent), and the a priori probability $P\{\bar{y}_i \mid x_i = \lambda_j\}$ for all $i \in V$ and $j \in \Lambda$, The problem of probabilistic relaxation labeling is to find a consistent label assignment for objects in V based on the a posteriori probability $P\{x_i = \lambda_j \mid Y\}$, the probability that assigns label λ_j to object i under observation set Y. x_i is the random variable defined for object i on label set Λ. Two objects connected by an edge are neighbors of each other. For object i, its *neighborhood* ∂i is defined as the set of all its neighbors, $\partial i = \{j \mid (i, j) \in E, j \neq i\}$. For object set V, *its neighborhood system* ∂V is the collection of neighborhoods for the objects in V. The label of an object highly relates to the labels of its neighbors (called *directly interacting objects*). Other objects provide indirect contextual information.

• *The authors are with the Department of Computer Science, Louisiana State University, Baton Rouge, LA 70803-4020. E-mail: iyengar@bit.csc.lsu.edu.*

Manuscript received March 1993; revised Feb. 13, 1996. Recommended for acceptance by K. Boyer.
For information on obtaining reprints of this article, please send e-mail to: transactions@computer.org, and reference IEEECS Log Number P96024.

For a given problem, $X = \{x_1, ..., x_n\}$ is the set of random variables for the object set V. $\omega = \{x_1 = \lambda_{j_1}, x_2 = \lambda_{j_2}, ..., x_n = \lambda_{j_n}\}$, called a configuration, represents a label assignment for the object set. The set of all configurations is called a configuration space $\Omega = \Lambda^{|X|}$. We use X_i to denote the set of all the random variables associated with objects in V excluding object i, $X_i = \{x_1, x_2, ..., x_{i-1}, x_{i+1}, ..., x_n\}$. ω_i denotes a configuration for X_i, and Ω_i is the corresponding configuration space. $\omega_{\partial i}$ represents a configuration for variable set $X_{\partial i} = \{x_j \mid j \in \partial i\}$, a neighborhood configuration for object i. And $\Omega_{\partial i}$ denotes the configuration space of configurations $\omega_{\partial i}$ over ∂i.

ω is regarded as a sample realization of a random field X over Ω. This random field X is called a *Markov Random Field* if the measures (or probabilities) P of its configurations ω have the following two properties:

1) $P\{\omega\} > 0$, for all $\omega \in \Omega$; and
2) the locality of Markov Random Field $P\{x_i = \lambda_j \mid \omega_i\} = P\{x_i = \lambda_j \mid \omega_{\partial i}\}$, i.e., the probability of assigning label λ_j to object i depends only on object i's neighborhood label configuration.

We consider random variable set X as a Markov Random Field with its neighborhood relation defined by graph G. According to property 2, $P\{x_i = \lambda_j \mid \omega_i\}$ can be estimated by $P\{x_i = \lambda_j \mid \omega_{\partial i}\}$.

The problem of relaxation labeling was described by Rosenfeld et al. [16]. Later, various approaches were developed that fall into two categories: discrete relaxation labeling and probabilistic relaxation labeling. In discrete relaxation labeling, label assignments are either possible or impossible. However, in probabilistic relaxation labeling, label assignments are measured by probabilities. Therefore, the construction of update functions over the probability vector space is a critical issue. Rosenfeld et al. concluded that nonlinear probabilistic update functions yielded the best results. However, their heuristic nonlinear update function induces the problem of bias, convergence, and choice of supporting function, etc. Hummel and Zucker [10] addressed some of these problems. They introduced a projected gradient update scheme and derived the property of local convergence for their update function. The function has a sound theoretical basis and permits an analytic proof of convergence. However, the update function is not easy to implement efficiently.

The effectiveness of relaxation labeling lies in its use of contextual information to eliminate ambiguous labels, which is achieved by iterative label assignment updates. Markov Random Field (MRF) theory provides a theoretical basis. Geman and Geman [3] considered images as instances of MRFs. Energy functions were defined for the MRFs such that the original image has minimal energy. A stochastic approach to minimize the energy utilized a simulated annealing technique and resulted in a highly parallel relaxation algorithm that uses the a posteriori distribution to yield a MAP estimate, restoring images from degraded observations. However, their method has a low convergence rate. Due to the nature of simulated annealing, hundreds of iterations are needed to obtain good restoration.

Pelkowitz [15] developed a probabilistic relaxation algorithm using MRF theory by applying the Maximum Entropy approach and deriving a multilinear relaxation update function. The function is data dependent and the final configuration (a fixed point in the configuration space) is a function of observations that locally optimizes the a posteriori probability.

Kittler and Hancock ([5] and [12]) developed an evidence combining formula in the framework of probability theory. They derived a nonlinear update function that is similar to Rosenfeld's.

Because no heuristics were used, the four problems that Rosenfeld encountered were solved. However, one of the difficulties was its potential computational complexity. The number of possible label configurations is exponential in the number of objects. In reality, far from being exponential, the number of permissible configurations is relatively small because labeling problems are highly structured. An exhaustive enumeration of permissible configurations is possible. This can significantly improve the efficiency of relaxation labeling. In this way, Kittler and Hancock successfully developed their algorithm. Their experimental results show better performance than some well-known edge detection algorithms like Canny's [1], Hildretch's [9], and Spacek's [18].

Theoretically, relaxation algorithms are computationally intensive and their convergence rates are still not well formulated. Therefore, improving their efficiency, especially under certain practical assumptions that hold for most applications, is an important goal. In this study, we investigate a new probabilistic relaxation labeling algorithm that derives from a different update function and a different dictionary construction method.

2 Our Approach

Given a set of objects V and a set of observations Y over V, after the a posteriori probability of assigning label λ_j to object i under Y, $P\{x_i = \lambda_j \mid Y\}$, is calculated for $i \in V$ and $\lambda_j \in \Lambda$, the label assignment with the highest a posteriori probability can be selected as the label assignment for object i, if $P\{x_i = \lambda_j \mid Y\} = max_{k=1}^{m} P\{x_k = \lambda_{j_k} \mid Y\}$.

However, relations among objects in V are usually complicated and not easy to model. Calculating $P\{x_i = \lambda_j \mid Y\}$ from observation Y directly is difficult. One way to solve this problem is to give $P\{x_i = \lambda_j \mid Y\}$ a reasonable estimate.

Let's first consider a simple estimate for $P\{x_i = \lambda_j \mid Y\}$. Assume that a label assignment for an object depends on the observations on that object only, $P\{x_i = \lambda_j \mid Y\} = P\{x_i = \lambda_j \mid \vec{y}_i\}$. When a priori probability $P\{x_i = \lambda_j\}$ and conditional probability $P\{\vec{y}_i \mid x_i = \lambda_j\}$ are both known, $P\{x_i = \lambda_j \mid Y\}$ can be estimated by

$$P\{x_i = \lambda_j \mid Y\} = P\{x_i = \lambda_j \mid \vec{y}_i\} = \frac{P\{\vec{y}_i \mid x_i = \lambda_j\} P\{x_i = \lambda_j\}}{P\{\vec{y}_i\}} \quad (1)$$

$$P\{\vec{y}_i\} = \sum_{j=1}^{m} P\{\vec{y}_i \mid x_i = \lambda_j\} P\{x_i = \lambda_j\} \quad (2)$$

But, $\vec{y}_i$ contains noise. When the above estimates are used to assign labels, errors will occur. The assumption that label assignments only depend on local observations presents the problem. Error labels are likely inconsistent with the label context. Thus, the label context can be used to refine the estimates. However, the interactions between objects are complicated. Capturing all the interactions with a single formula is impractical. In order to incorporate contextual information into the estimates, the neighborhood system ∂V is used to divide the interactions into two categories: direct interactions with the neighbors and indirect interactions with other objects. Using direct interactions to estimate $P\{x_i = \lambda_j \mid Y\}$ balances the need to use contextual information and the complexity of modeling object interactions.

In order to formulate direct interactions, we assume that a label assignment only depends on the label configuration of its neighboring objects and its observations $\vec{y}_i$. This makes label assignment X a MRF. Under this assumption, we have Lemma 1. Lemma 1 is deduced from Pelkowitz's Lemma 1 [15] with Bayes' rule. We call (3), (4), and (5) update function $\mathcal{T}$.

LEMMA 1.

$$P^{(r+1)}\{x_i = \lambda_j\} = \sum_{\omega_{\partial i} \in \Omega_{\partial i}} W_{\omega_{\partial i}}^{(r)} E^{(r)}(i, j, \omega_{\partial i}) \quad (3)$$

where

$$E^{(r)}(i, j, \omega_{\partial i}) = \frac{P^{(r)}\{x_i = \lambda_j\} P\{\omega_{\partial i} \mid x_i = \lambda_j\}}{\sum_{k=1}^{m} P^{(r)}\{x_i = \lambda_k\} P\{\omega_{\partial i} \mid x_i = \lambda_k\}} \quad (4)$$

$$W_{\omega_{\partial i}}^{(r)} = \prod_{x_i = \lambda_{j_i} \in \omega_{\partial i}} P^{(r)}\{x_i = \lambda_{j_i}\} \quad (5)$$

Based on $\mathcal{T}$, our algorithm is as follows. First, vector $P^{(0)}$ in the mn dimensional probability vector space is initialized by (1) and (2). Then, $\mathcal{T}$ is recursively applied to vector $P^{(r)}$, $r \geq 0$ to obtain a new vector $P^{(r+1)}$ which provides label assignment estimates over a gradually expanding contextual area. When the update process finally converges, the label assignment is found by maximizing the a posteriori probabilities.

The time complexity of the algorithm, in the worst case, is exponential in the number of objects. However, this complexity can be reduced considerably in two ways. First, the sizes of the neighborhoods are usually much smaller than the size of the object set. In image processing, neighborhood sizes are usually 3×3 or 5×5. However, even for a 3×3 neighborhood, the total number of configurations is m^9, m being the size of label set Λ. In reality, applications are well-structured and most configurations are physically impossible, so the number of configurations which can occur in a given application is relatively small.

2.1 Types of Configurations

Besides permissible configurations, there are other configurations that have significant influence on label assignments. For example, in edge detection, a pixel i takes its eight surrounding pixels as its neighboring pixels (See Fig. 1a). All the possible one pixel edge patterns across this 3×3 area are permissible configurations. One permissible configuration in this setting is showed in Fig. 1b. Here, arrows indicate the directions of edge pixels and blanks indicate nonedge pixels. The label set contains five elements: the four different edge directions $\rightarrow$, $\uparrow$, $\leftarrow$, and $\downarrow$, and a nonedge label ϵ. The permissible configuration in Fig. 1b is denoted by $\{x_0 = \epsilon, x_1 = \epsilon, x_2 = \epsilon, x_3 = \epsilon, x_4 = \uparrow, x_5 = \epsilon, x_6 = \uparrow, x_7 = \uparrow, x_i = \epsilon\}$ and is associated with probability $P\{x_0 = \epsilon, x_1 = \epsilon, x_2 = \epsilon, x_3 = \epsilon, x_4 = \uparrow, x_5 = \epsilon, x_6 = \uparrow, x_7 = \uparrow \mid x_i = \epsilon\}$.

Now, let's assign another label for x_i in the same neighborhood setting, for example, $x_i = \uparrow$ (Fig. 1c). It is possible that in the previous relaxation iterations, pixel 6 is erroneously labeled. Its label should be ϵ, not $\uparrow$. Because this error may be corrected later in the relaxation process, the configuration provides support to assign $\uparrow$ to pixel i as suggested by the permissible configuration in Fig. 1d. Thus, there are configurations that may contribution to label assignments and that are not permissible configurations. We call them *possible supporting configurations*.

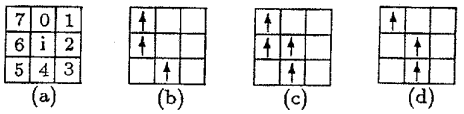


Fig. 1. Possible supporting configurations.

In our scheme, a dictionary contains two kinds of configurations: *permissible configurations* that occur in ideally labeled situations, and those configurations which would lead to permissible configurations if some of the neighboring object labels change while the label for object i remains the same. We call these configurations *possible supporting configurations*.

2.2 Dictionary Schemes

Kittler and Hancock ([5], [6], [7], [12]) proposed a dictionary construction method. Each object i has a dictionary D_i constructed from all permissible configurations of object i's neighborhood ∂i. Dictionary D_i is further divided into m sections $D_i(\lambda_j)$ according to different label assignments for object i, $x_i = \lambda_j$, $j = 1, \ldots, m$. Let $\lambda_{j_l}^k$ be the label on object l, $l \neq i$ of the kth configuration in section $D_i(\lambda_j)$. The kth configuration in $D_i(\lambda_j)$ is denoted as

$$I_i^k(\lambda_j) = \left\{x_i = \lambda_j, x_l = \lambda_{j_l}^k, l \in \partial i\right\}.$$

Associated with every permissible configuration is the probability $P\{x_i = \lambda_j, x_l = \lambda_{j_l}^k, l \in \partial i\}$. For physically impossible configurations, i.e., $\{x_i = \lambda_j, x_j = \lambda_{j_l}^k, l \in \partial i\} \notin D_i(\lambda_j)$, the probabilities are set to be zero.

In update function $\mathcal{T}$, all the probabilities are either the a priori probabilities that remain unchanged during the relaxation process, or the a posteriori probability estimates $P^{(r)}\{x_i = \lambda_j \mid Y\}$ that are obtained from the previous iteration. So, it is better to associate with each dictionary item the a priori probability

$$P\{x_j = \lambda_{j_l}^k, l \in \partial i \mid x_i = \lambda_j\}.$$

A dictionary with only permissible configurations is not suitable for our relaxation scheme. Equations (6) and (7) provide the update function for Kittler and Hancock's algorithm. For each object i, (7) is used to sum the support for a label assignment λ_j from all permissible configurations, and hence to obtain a supporting function value Q for that label. Normalization is performed by (6) after supports for all labels are found. A dictionary of permissible configurations works for this scheme.

Using our update function $\mathcal{T}$, normalizations for all possible label assignments are performed for each configuration. Then, the normalized supports from all the configurations for a particular label assignment are added together. Normalization insures a proper distribution of support from a neighborhood setting to all possible label assignments. However, in many applications, most neighborhood settings have no, or only one, permissible configuration. This makes the distribution of support by (4) inefficient, and so deteriorates the relaxation process. For example, in the edge detection problem we mentioned above, with those 165 configurations, there are 149 neighborhood settings. Only 16 of them have two permissible configurations. Others have only one. Thus, the dictionary constructed by Kittler and Hancock's method does not work well with $\mathcal{T}$.

$$P^{(r+1)}\left\{x_i = \lambda_j\right\} = \frac{P^{(r)}\left\{x_i = \lambda_j\right\}Q^{(r)}\left\{x_i = \lambda_j\right\}}{\sum_{k=1}^{m} P^{(r)}\left\{x_i = \lambda_k\right\}Q^{(r)}\left\{x_i = \lambda_k\right\}} \tag{6}$$

where

$$Q^{(r)}\left\{x_i = \lambda_j\right\} = \frac{1}{P\left\{x_i = \lambda_j\right\}} \sum_{\omega_{\partial i} \in \Omega_{\partial i}} \left[\left\{ \prod_{(x_k = \lambda_{j_k}) \in \omega_{\partial i}} \frac{P^{(r)}\left\{x_k = \lambda_{j_k}\right\}}{P\left\{x_k = \lambda_{j_k}\right\}} \right\} P\left\{x_i = \lambda_j, \omega_{\partial i}\right\} \right] \tag{7}$$

A suitable dictionary for our scheme contains all the configurations for the neighborhood settings that are obtained from permissible configurations. In the example of edge detection, each neighborhood setting contains five different configurations. In all, the new dictionary, called D_i', has 745 configurations of which 165 are permissible configurations while the rest are possible supporting configurations.

Dictionary D_i' is a table with s rows and m columns, where s is the number of neighborhood settings and m is the size of the label set. $D_i'(\lambda_j)$ is a column corresponding to the assignment of label λ_j to object i. Let $\lambda_{j_l}^k$ be the label on object l, $l \neq i$, of the kth neighborhood configuration in column $D_i'(\lambda_j)$. The kth configuration in $D_i'(\lambda_j)$ is $C_i^k(\lambda_j) = \{x_l = \lambda_{j_l}^k, l \in \partial i \mid x_i = \lambda_j\}$, and the associated probability is $P\{x_l = \lambda_l^k, l \in \partial i \mid x_i = \lambda_j\}$, the probability that configuration $\{x_i = \lambda_j, x_l = \lambda_{j_l}^k, l \in \partial i\}$ occurs when $x_i = \lambda_j$.

The new dictionary can be derived from K & H's. First, we calculate the a priori probability $P\{x_i = \lambda_j\}$ by adding together the probabilities of all permissible configurations in $D_i(\lambda_j)$:

$$P\left\{x_i = \lambda_j\right\} = \sum_{I_i^l(\lambda_j) \in D_i(\lambda_j)} P\left\{I_i^l(\lambda_j)\right\}.$$

Then, the probability for each permissible configuration in $D_i'(\lambda_j)$ is obtained by

$$P\left\{C_i^k(\lambda_j)\right\} = \frac{P\left\{I_i^k(\lambda_j)\right\}}{P\left\{x_i = \lambda_j\right\}}.$$

The probabilities for possible supporting configurations are calculated from the permissible configurations. possible supporting configurations occurs when labeling errors are presented. Thus, it is natural to consider possible supporting configurations as corrupted permissible configurations. In [7], Kittler and Hancock proposed a label error process for discrete relaxation. They derived formulas to estimate the probability for any configuration from permissible configurations. The idea is to sum the likelihoods of the configuration with all the permissible configurations. We adopt this method to estimate the probabilities for possible supporting configurations. Assuming that labeling errors occur with equal probability p_e, the likelihood of a possible supporting configuration $\{x_i = \lambda_j, \omega_{\partial i}\}$ with a permissible configuration $I_i^k(\lambda_j) \in D_i(\lambda_j)$ is calculated by

$$P\left\{x_i = \lambda_j, \omega_{\partial i} \mid I_i^k(\lambda_j)\right\} = (1 - p_e)^{|\partial i| - K(i,k)} p_e^{K(i,k)} \tag{8}$$

Now, $|\partial i|$ is the number of neighborhood objects and $K(i, k)$ is the number of labels that are different between a configuration $\omega_{\partial i}$ and a given permissible configuration $I_i^k(\lambda_j)$. This likelihood is called the *neighborhood transition probability*. The probability of a possible supporting configuration is the summation of the neighborhood transition probabilities over all permissible configurations.

$$P\left\{x_i = \lambda_j, \omega_{\partial i}\right\} = \sum_{I_i^l(\lambda_j) \in D_i(\lambda_j)} P\left\{x_i = \lambda_j, \omega_{\partial i} \mid I_i^k(\lambda_j)\right\} P\left\{I_i^l(\lambda_j)\right\} \tag{9}$$

To summarize, the procedure of our dictionary construction is as follows:

1) Find all permissible configurations in the application and assign each permissible configuration a probability;
2) Find all possible supporting configurations from the set of permissible configurations;
3) Use label error process, (8) and (9), to calculate probabilities for all possible supporting configurations;
4) The permissible configurations and the possible supporting configurations together form the dictionary D_i'.

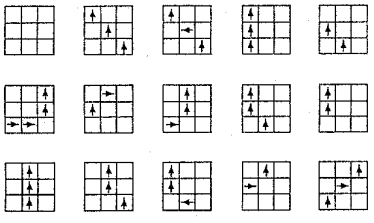
2.3 An Edge Detection Application

Edge detection is very important in image processing. The performance of many vision systems depend on the performance of their edge detectors. Traditional edge detectors use first or second order derivatives. However, these detectors are very sensitive to

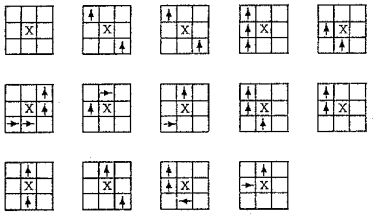
noise. The noise characteristics of an edge detector depend on the size of the operator. Though larger sized operators reduce more random noise, they are also more likely to simultaneously lose some important edge features, degrading their resolutions.

Using relaxation labeling as a postprocessing method is one prospective solution. First, a small size edge detector is employed to obtain an initial edge assignment for every pixel. Then, a dictionary of permissible configurations and possible supporting configurations in the 3×3 neighborhood of each pixel is constructed and a probabilistic relaxation labeling algorithm is used to remove erroneously labeled pixels. This postprocessing reduces noise and helps to produce one pixel wide edge descriptions. In edge detection, the object set $V = \{(u, v) \mid (u, v)$ is the position of a pixel in the given image$\}$. For each pixel (u, v), vector $\bar{y}_{(u,v)}$ consists of two first order partial differences cu and cv of the observed noisy intensity $g'(u, v)$, $c_u = g'(u + 1, v) - g'(u, v)$, and $c_v = g'(u, v + 1) - g'(u, v)$. The additive noise is assumed to be Gaussian distributed with a zero mean and a standard deviation of σ, independent of the underlying noise-free image intensity $g(u, v)$. The noise variance σ is estimated from the statistics of the given image (Kittler et al. [14]). Hancock and Kittler [5] described how to estimate initial label probabilities $P\{x = \epsilon \mid c_u, c_v\}$, $P\{x = \rightarrow \mid c_u, c_v\}$, $P\{x = \leftarrow \mid c_u, c_v\}$, $P\{x = \uparrow \mid c_u, c_v\}$, and $P\{x = \downarrow \mid c_u, c_v\}$.

In Section 2.2, we detailed how to construct the dictionary. A permissible configuration in a 3×3 lattice has only one continuous, single pixel wide edge. The 165 permissible configurations are found by manipulating the 15 configurations shown in Fig. 2a by reversal, reflection and rotation. 97 of them have label e for the center pixel, with 17 for each of the four different directional edge labels. In this study, all permissible configurations are considered equally likely. Out of these 165 configurations, 149 neighborhood settings are found. All of them are from the reversal, reflection and rotation of the 14 neighborhood settings in Fig. 2b.



(a) Permissible configurations.



(b) Neighborhood settings.

Fig. 2. Permissible configurations and neighborhood settings for edge detection. Empty cells indicate nonedge pixels. "X" for any of the five labels.

Our relaxation approach differs from Pelkowitz¢s approach in two aspects. Although we both use maximum entropy estimate for the joint conditional probability $P\{\omega_{\mathcal{N}_i} \mid Y\}$, Pelkowitz¢s update function is multilinear; ours is nonlinear. Another difference is the implementation method. Pelkowitz proposed to divide the set of configurations into two:

$$P^{(s+1)}\{x_i = \lambda_j\} = \sum_{\omega \in \Omega_i^C} W_{\omega|Y} E(i, j, \omega) + \sum_{\omega \in \Omega_i^D} W_{\omega|Y} E(i, j, \omega)$$

Ω_i^D, the "don't care" configuration set, contains configurations such that $P\{x_i = \lambda_k\} = P\{x_i = \lambda_k \mid \omega\}$ for all λ_k. In this setting, $P\{x_i = \lambda_j\}$ is independent of its neighbors. Ω_i^C, the "care" configuration set, is the complement of Ω_i^D. Configurations in Ω_i^C are in favor of changing the has about 3×10^6 configurations. If $|\Omega| >> |\Omega_i^D|$, i.e., $|\Omega| \approx |\Omega_i^C|$, the computation is very expensive because there are large number of configurations to be considered. If $|\Omega| >> |\Omega_i^C|$, which means that only a few "care" configurations are selected, the expense to compute $\mathcal{T}$ is acceptable. However, the convergence rate will be very slow because the factor for change ($\Sigma_{\omega \in \Omega_i^C}$) is much smaller than the factor for stability ($\Sigma_{\omega \in \Omega_i^D}$). In our implementation, the use of a configuration dictionary avoids this difficulty. Experiments show that our algorithm converges after 10 iterations in most cases.

3 Experimental Results

The edge detection algorithm described above has been implemented along with two versions of Kittler and Hancock's algorithm. One uses their original dictionary, while the other uses our new dictionary. The behaviors of them are examined on both synthetic images and natural pictures. The two templates that are used to compute the first order differences are the smallest among the existing templates (2×1 and 1×2); they preserve more edge pixels and also keep more noise pixels. We choose these small templates because they can test the noise insensitivity and the robustness of relaxation algorithms to the greatest extent, and we can compare our algorithm with Kittler and Hancock's algorithm in an identical situation.

An important feature for both algorithms is the convergence rate. After 10 iterations, the algorithms essentially converge and most pixel¢s probabilities are either zero or one. In the following presentation, all the edge outputs are collected after 10 iterations. The figures show the final maximum a posteriori label assignments. No postprocessings such as linking, thinning, or cleaning, are used.

We tested the algorithms on a well structured 50×50 synthetic image (Fig. 3). Within the circle, the gray intensity is 56 (the range of gray level is [0, 255]). Inside the square, the gray intensity is 231. Outside the square, the gray intensity is 115. This image is then mixed with independent Gaussian noise of a zero mean and different standard deviations of 40, 80, 120, 160, or 200. The experiment tests the performance of the algorithms under different noise levels, and the ability of the algorithms to detect edges of various orientations and curvatures.

Fig. 3b is the result obtained from our algorithm, Fig. 3c from Kittler and Hancock's algorithm with their own dictionary, and Fig. 3d from Kittler and Hancock's algorithm with the new dictionary. All three algorithms work fine for both lines and curves in all orientations. All the corners are well reconstructed and the edges are continuous. Regarding noise suppression, our algorithm is better than Kittler and Hancock's algorithm. It eliminates much of the noise effects and gets good results for images with noise standard deviation less than 100. Reasonable results are obtained for noise levels up to $\sigma = 200$. Kittler and Hancock's algorithm with its own dictionary works fine for noise levels of $\sigma < 60$, but after this range the performance deteriorates. Kittler and Han-

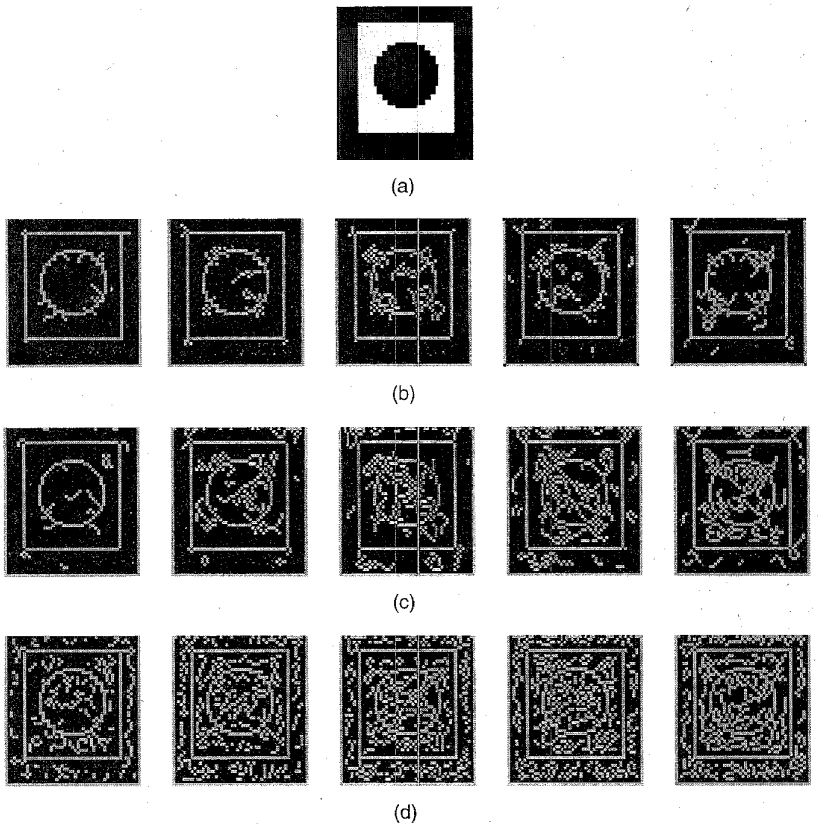


Fig. 3. Synthetic image (0-255) with additional Gaussian noise (σ in 20, 40, 60, 80, 100, 120, 140, 180, and 200). (a) is the original image; (b) is the result from our algorithm; (c) is from Kittler and Hancock's method with dictionary D_i; (d) is from Kittler and Hancock's method with dictionary D_i'.

cock's algorithm using the new dictionary works fine only when the additional noise has a standard deviation of no greater than 20. Otherwise, too many noise pixels are preserved.

Color pictures are also used to test the algorithms. The intensity is calculated by averaging the intensities of red, green, and blue. Fig. 4a (256 × 256) tests the ability of the algorithms to identify edges in the existence of texture. All three methods succeed in obtaining edges of the house with good connectedness, even for the very sharp curves. Among these three methods, the edge connectedness in Fig. 4d is the best, and that in Fig. 4b is next. However, both Fig. 4b and Fig. 4c eliminate texture effects better than Fig. 4d. In Fig. 5 (450 × 320), Kittler and Hancock's original algorithm fails to get edges for the simple patterns on the wall. Again, Kittler and Hancock's algorithm with our new dictionary obtains the best edge connectedness but retains many noise pixels. Our algorithm achieves a better balance between making noticeable edges and reducing noise. Fig. 6 (450 × 320) shows the same performance differences among these three implementations.

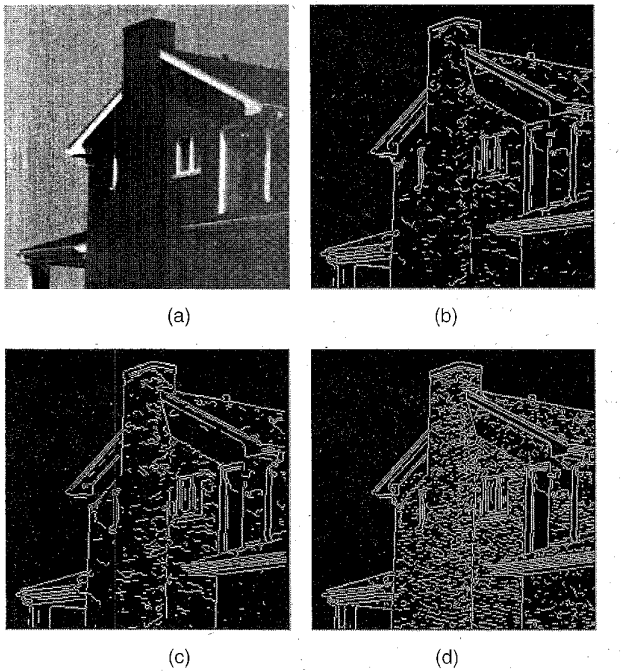


Fig. 4. Image of a house. (a) is the original input image; (b) is the output from our algorithm; (c) is the output from Kittler and Hancock's algorithm with dictionary D_i; (d) is the output from Kittler and Hancock's algorithm with dictionary D_i'.

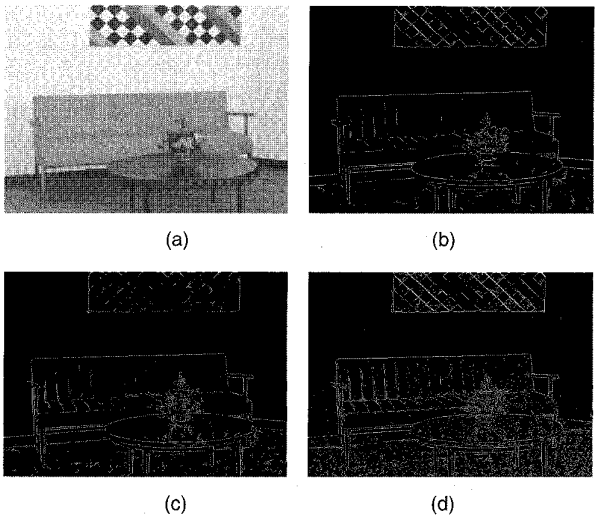


Fig. 5. The image of an office. (a) is the original input image; (b) is the output from our algorithm; (c) is the output from Kittler and Hancock's algorithm with dictionary D_i; (d) is the output from Kittler and Hancock's algorithm with dictionary D_i'.

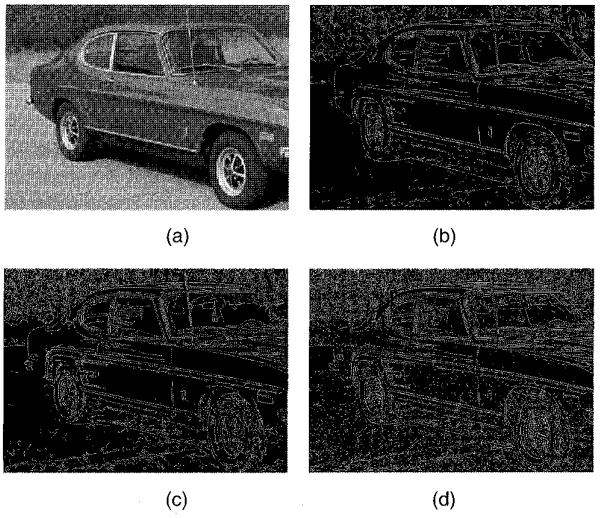


Fig. 6. The image of a car. (a) is the original input image; (b) is the output from our algorithm; (c) is the output from Kittler and Hancock's algorithm with dictionary D_i; (d) is the output from Kittler and Hancock's algorithm with dictionary D_i'.

From these experiments, we observe that our relaxation algorithm is quite successful in using contextual information to perform postprocessing for edge detection. It achieves a good balance between preserving edge features and eliminating noise effects. However, in some places, the edge output is not well connected. One explanation is that, although using contextual information through relaxation can eliminate ambiguities from imprecise initial label assignments, if the initial assignment contains too many labeling errors, label contextual information may not be enough to correct all of them.

4 Conclusion

In this paper, we presented a new probabilistic relaxation algorithm based on a dictionary construction method. This algorithm is then successfully used in edge detection. The experiments show that our relaxation edge detector converges quickly and successfully uses contextual information to balance edge preservation and noise suppression.

Compared to Kittler and Hancock's approach, the difference is in the order of normalization and evidence combining. Theirs combines all the evidence from all permissible configurations to get a supporting function value Q first. Ours first distributes the support of a configuration among the label assignments for an object, then adds all the support together. Furthermore, a new method is used to construct the configuration dictionary. This new dictionary contains more configurations.

Acknowledgments

The authors thank the anonymous reviewers for their helpful suggestions and comments. We acknowledge special thanks to the editor, Dr. Kim Boyer and to Mr. Hla Min for excellent proofreading of this paper.

This project was partly funded by the U.S. Department of the Navy, N00014-92-K-6002.

References

[1] J. Canny, "Computational Approach to Edge Detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 699–714, 1986.

[2] L.S. Davis, C.Y. Wang, and H.C. Xie, "An Experiment in Multispectral, Multitemporal Crop Classification Using Relaxation Techniques," *Computer Vision, Graphics, and Image Processing*, vol. 23, pp. 227–235, 1983.

[3] S. Geman and D. Geman, "Stochastic Relaxation, Gibbs Distributions and Bayesian Restoration of Images," *IEEE Pattern Analysis and Machine Intelligence*, vol. 6, pp. 721–741, 1984.

[4] J. Gu, W. Wang, and T.C. Henderson, "A Parallel Architecture for Discrete Relaxation Algorithm," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 9, no. 6, pp. 816–831, 1987.

[5] E.R. Hancock and J. Kittler, "Edge-Labeling Using Dictionary-Based Relaxation," *IEEE Trans Pattern Analysis and Machine Intelligence*, vol. 12, no. 2, pp. 165–181, 1990.

[6] E.R. Hancock and J. Kittler, "Discrete Relaxation," *Pattern Recognition*, no. 23, pp. 711–733, 1990.

[7] E.R. Hancock and J. Kittler, "A Label Error Process for Discrete Relaxation," *IEEE 10th ICPR*, vol. 1, pp. 523–528, 1990.

[8] F.R. Hansen and H. Elliott, "Image Segmentation Using Simple Markov Random Fields," *Computer Graphics and Image Processing*, vol. 20, pp. 101–132, 1982.

[9] J. Hilditch, "Linear Skeletons Form Square Cupboards," *Machine Intelligence*, vol. 6, pp. 403–420, 1969.

[10] R.A. Hummel and S.W. Zucker, "On the Foundations of Relaxation Labeling Processes," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 5, pp. 267–287, 1983.

[11] J. Kittler and J. Foglein, "On Compatibility and Support Functions in Probabilistic Relaxation," *Computer. Vision, Graphics, and Image Processing*, 1986.

[12] J. Kittler and E.R. Hancock, "Combining Evidence in Probabilistic Relaxation," *Int'l J. Pattern Recognition Artificial Intelligence*, vol. 3, pp. 29–52, 1989.

[13] J. Kittler and J. Illingworth, "Relaxation Labeling Algorithms—A Review," *Image and Vision Computers*, vol. 3, no. 4, pp. 206–216, 1985.

[14] J. Kittler, J. Illingworth, J. Foglein, and K. Paler, "An Automatic Thresholding Algorithm and Its Performance," *Proc. Seventh ICPR*, Montreal, 1984.

[15] L. Pelkowitz, "A Continuous Relaxation Labeling Algorithm for Markov Random Fields," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 20, no. 3, pp. 709–715, 1990.

[16] A. Rosenfeld, R.A. Hummel, and S.W. Zucker, "Scene Labeling by Relaxation Operations," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 6, pp. 420–433, 1976.

[17] B.J. Schachter, A. Lev, S.W. Zucker, and A. Rosenfeld, "An Application of Relaxation Methods to Edge Reinforcement," *IEEE Trans. Systems, Man and Cybernetics*, vol. 7, pp. 813–816, 1977.

[18] L.A. Spacek, "Edge Detection and Motion Detection," *Image and Vision Computing*, vol. 4, no. 1, pp.. 43–56, Feb. 1986.