CHAPTER 13

MORPHOLOGICAL HYPERSPECTRAL IMAGE CLASSIFICATION: A PARALLEL PROCESSING PERSPECTIVE

ANTONIO J. PLAZA
Department of Computer Science, University of Extremadura, E-10071 Caceres, Spain

13.1. INTRODUCTION

Mathematical morphology (MM) is a theory for spatial structure analysis that was established by introducing fundamental operators applied to two sets [1]. A set is processed by another one having a carefully selected shape and size, known as the structuring element (SE). In the context of image processing, the SE acts as a probe for extracting or suppressing specific structures of the image objects, checking that each position of the SE fits within those objects. Based on these ideas, two fundamental operators are defined in MM, namely *erosion* and *dilation*. The application of the erosion operator to an image yields an output image, which shows where the SE *fits* the objects in the image. On the other hand, the application of the dilation operator to an image produces an output image, which shows where the SE *hits* the objects in the image. All other MM operations can be expressed in terms of erosion and dilation [2]. For instance, the notion behind the *opening* operator is to dilate an eroded image in order to recover as much as possible of the eroded image. In contrast, the *closing* operator erodes a dilated image so as to recover the initial shape of image structures that have been dilated. The filtering properties of the opening and closing are based on the fact that, depending on the size and shape of the considered SE, not all structures from the original image will be recovered when these operators are applied. Because of the nonlinear properties of MM filters, their application generally results in an irreversible, though controlled, loss of information.

Although MM operators were originally defined for binary images, they were soon extended to gray-tone (mono-channel) images by viewing these data as an imaginary topographic relief in which the brighter the gray tone, the higher the corresponding elevation [3]. Here, morphological operations can be graphically

Hyperspectral Data Exploitation: Theory and Applications, Edited by Chein-I Chang

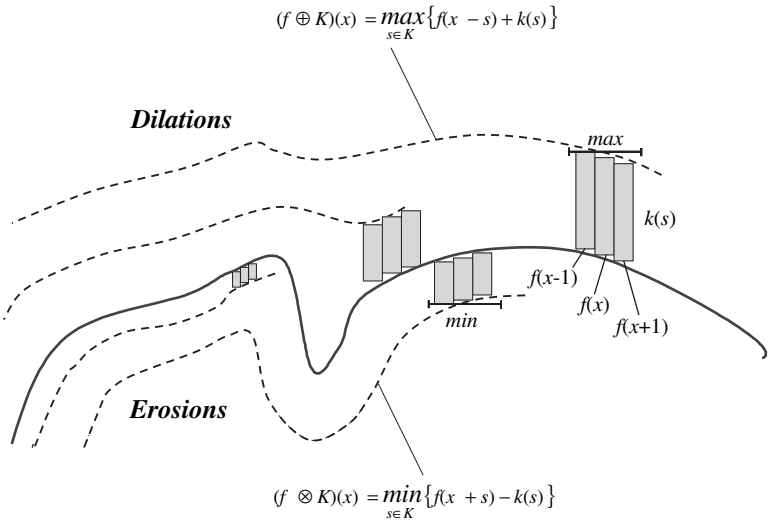

Figure 13.1. Graphical interpretation of grayscale morphological erosion and dilation operations.

interpreted as the result of sliding a flat SE over the topographical relief, so that the SE defines the new (dilated or eroded) scene based on its spatial properties such as height or width (see Figure 13.1). However, extension of MM operators to multichannel data such as hyperspectral imagery with hundreds of spectral channels is not straightforward. A simple approach consists in applying grayscale MM techniques to each channel separately, an approach that has been called marginal MM in the literature [4]. However, the marginal approach is often unacceptable in remote sensing applications because, when MM techniques are applied independently to each image channel, analysis techniques are subject to the well-known problem of ''false colors''; that is, it is very likely that new spectral constituents not present in the original image may be created as a result of processing the channels separately [5]. An alternative way to approach the problem of multichannel MM is to treat the data at each pixel as a vector. Unfortunately, there is no unambiguous means of defining the minimum and maximum values between two vectors of more than one dimension, and thus it is important to define an appropriate arrangement of vectors in the selected vector space.

To be able to define appropriate vector ordering schemes for remote sensing-driven applications, it is important to take into account the requirements of available techniques for analyzing data of interest. In particular, the special characteristics of hyperspectral images pose different processing problems, which must be necessarily tackled under specific mathematical formalisms, such as classification, segmentation, spectral unmixing, and so on. A diverse array of techniques has been applied to extract information from hyperspectral data during the last decade. They are inherently either full pixel techniques or mixed pixel techniques, where each pixel vector in a hyperspectral scene provides a ''spectral signature'' that uniquely

characterizes the underlying materials at each site in a scene. The underlying assumption governing full pixel techniques is that each pixel vector measures the response of one single material. In contrast, the underlying assumption governing mixed pixel techniques (also called ''spectral unmixing'' approaches) is that each pixel vector measures the response of multiple underlying materials at each site. A hyperspectral scene (sometimes referred to as ''data cube'') is often a combination of the two situations, where a few sites in a scene are pure materials, but many other are mixtures of materials. Most available techniques for hyperspectral data processing focus on analyzing the data without incorporating information on the spatially adjacent data; that is, the hyperspectral data are treated not as an image but as an unordered listing of spectral measurements. It is worth noting that such spectral-based techniques would yield the same result for a data cube, as well as for the same data cube where the spatial positions have been randomly permuted. However, one of the distinguishing properties of hyperspectral data is the multivariate information coupled with a two-dimensional (pictorial) representation amenable to image interpretation. Subsequently, there is a need to incorporate the spatial component of the data in the development of techniques for hyperspectral data exploitation. As will be shown in this chapter, mathematical morphology offers a remarkable framework to achieve the desired integration of spatial and spectral information in hyperspectral data analysis.

While integrated spatial/spectral developments hold great promise for hyperspectral image analysis, they also introduce new processing challenges. In particular, the price paid for the wealth of spatial and spectral information available from hyperspectral sensors is the enormous amounts of data that they generate. Several applications exist, however, where having the desired information calculated in near real time is a requirement. Such is the case of automatic target recognition for military and defense/security deployment. Other relevant examples include (a) environmental monitoring and assessment, (b) urban planning and management studies, and (c) risk/hazard prevention and response including wild-land fire tracking, biological threat detection, and monitoring of oil spills and other types of chemical contamination. With the recent explosion in the amount and complexity of hyperspectral imagery, parallel processing has soon become a tool of choice in many remote sensing missions, especially with the advent of low-cost systems such as commodity clusters and distributed networks of workstations.

The main goal of this chapter is to provide a seminal view on recent, consolidated advances in morphological techniques for efficient processing of hyperspectral imagery. Our main focus is on the development of joint spatial/spectral developments, able to exploit knowledge about the spatial arrangement of objects in the scene and to take advantage of the wealth of spectral information present in the data. Two new algorithms are developed, based on different strategies for ordering pixel vectors in spectral space. The considered approaches are (i) a supervised mixed pixel classification algorithm that naturally integrates both spatial and spectral information simultaneously and (ii) a morphological watershed-based algorithm that segments a hyperspectral scene in fully unsupervised fashion by first exploiting the spectral information and then making use of spatial context.

The chapter is structured as follows. Section 13.2 provides several vector ordering strategies used in this work to extend morphological operations to high-dimensional spaces. Section 13.3 develops two new algorithms for morphological classification of hyperspectral image data sets. Section 13.4 provides parallel processing support for the two algorithms above. An evaluation of the proposed techniques is then provided, along with a comparison to other existing approaches, using real hyperspectral data sets collected by the 224-channel NASA's Jet Propulsion Laboratory Airborne Visible-Infrared Imaging Spectrometer (AVIRIS) system. Parallel performance results are given in a massively parallel Beowulf cluster. Our final section concludes with some summarizing points and hints at plausible future research.

13.2. VECTOR ORDERING STRATEGIES FOR MULTIDIMENSIONAL MORPHOLOGICAL OPERATIONS

This section first provides an overview of available approaches for vector ordering in different applications, along with a detailed discussion on the appropriateness of using such strategies in the context of hyperspectral imaging applications. Then, it describes our adopted vector ordering strategy and briefly illustrates its advantages and disadvantages.

13.2.1. Available Approaches

Several vector ordering schemes have been discussed in the literature [4]. The choice between the different available options is generally application-driven. Four classes of ordering methods (marginal, reduced, partial, and conditional ordering) will be shortly illustrated here in the context of hyperspectral imaging applications. Let us first consider a hyperspectral image $\boldsymbol{f}$, defined on the N-dimensional space, where N is the number of spectral channels. Let $\boldsymbol{f}(x,y)$ and $\boldsymbol{f}(x',y')$ denote two pixels vectors at spatial locations (x,y) and (x',y'), respectively, with $\boldsymbol{f}(x,y) = [f_1(x,y),\ldots,f_N(x,y)]^T$ and $\boldsymbol{f}(x',y') = [f_1(x',y'),\ldots,f_N(x',y')]^T$. In marginal ordering (M-ordering), each pair of observations $f_i(x,y)$ and $f_i(x',y')$ would be ordered independently along each of the N channels [4]. In reduced ordering (R-ordering), a scalar parameter function g would be computed for each pixel of the image, and the ordering is performed according to the resulting scalar values. The ordered vectors would satisfy the relationship $\boldsymbol{f}(x,y) \leq \boldsymbol{f}(x',y') \Rightarrow g[\boldsymbol{f}(x,y)] \leq g[\boldsymbol{f}(x',y')]$. In partial ordering (P-ordering), the input multivariate samples would be partitioned into smaller groups, which would then be ordered. Both R-ordering and P-ordering may lead to the existence of more than one suprema (or infima) and, thus, introduce ambiguity in the resulting data. In conditional ordering (C-ordering), the pixel vectors would be initially ordered according to the ordered values of one of their components—for example, the first component, $f_1(x,y)$ and $f_1(x',y')$. As a second step, vectors with the same value for the first component would be ordered according to the ordered values of another component—for example, the second component, $f_2(x,y)$ and $f_2(x',y')$, and so on. This type of

ordering is not generally appropriate for hyperspectral data, where each spectral feature *as a whole* contains relevant information about the optical and physical properties of the observed land cover. In addition, pixel vectors in remote sensing are usually affected by atmospheric and illumination interferers, which may introduce fluctuations in the amount of energy collected by the sensor at the different wavelength channels. The incident signal is electromagnetic radiation that originates from the sun and is measured by the sensor after it has been reflected upwards by materials on the surface of the Earth. As a result, two differently illuminated pixels that belong to the same spectral constituent may be ordered inconsistently by the C-ordering and M-ordering schemes.

13.2.2. Proposed Vector-Ordering Strategy

In this subsection, we develop an application-driven vector ordering technique based on a spectral purity-based criterion, where each pixel vector is ordered according to its spectral distance to other neighboring pixel vectors in the data. This type of ordering, which can be seen as a modification of the D-ordering available in the literature [5], is based on the definition of a cumulative distance $D_B[\boldsymbol{f}(x,y)]$ between one particular pixel $\boldsymbol{f}(x,y)$, where $\boldsymbol{f}(x,y)$ denotes the N-dimensional vector at discrete spatial coordinates $(x,y) \in Z^2$, and all the pixel vectors in the spatial neighborhood given by B (B-neighborhood):

$$D_B[\boldsymbol{f}(x,y)] = \sum_s \sum_t \text{Dist}[\boldsymbol{f}(x,y), \boldsymbol{f}(s,t)], \qquad \forall (s,t) \in Z^2(B) \tag{13.1}$$

where Dist is a linear pointwise distance measure between two N-dimensional vectors. As a result, $D_B[\boldsymbol{f}(x,y)]$ is given by the sum of Dist scores between $\boldsymbol{f}(x,y)$ and every other pixel vector in the B-neighborhood. To be able to define the usual MM operators in a complete lattice framework, we need to be able to define a supremum and an infimum given an arbitrary set of vectors $S = \{\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_n\}$, where n is the number of vectors in the set. This can be achieved by computing $D_B(S)$ and selecting $\boldsymbol{s}_p$ such that $D_B[\boldsymbol{s}_p]$ is the minimum of that set, with $1 \leq p \leq n$. In similar fashion, we can select $\boldsymbol{s}_k$ such that $D_B[\boldsymbol{s}_k]$ is the maximum of that set, with $1 \leq k \leq n$. The selection of a maximum pixel vector making use of a pointwise distance measure Dist is graphically illustrated in Figure 13.2, where four pixels (vectors) respectively denoted by $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{c}$, and $\boldsymbol{d}$ are ordered by calculating their respective D_B-based scores and selecting the one that results in the maximum score. In the figure, we assume that the four pixel vectors above belong to the same spatial neighborhood given by a SE denoted by B.

Based on the simple definitions above, the extended erosion of $\boldsymbol{f}$ by B is based on the selection of the B-neighborhood pixel vector that produces the minimum value for D_B:

$$\begin{aligned}(\boldsymbol{f} \ominus B)(x,y) &= \{\boldsymbol{f}(x+s', y+t'), (s',t') \\ &= \arg\min_{(s,t) \in Z^2(B)} \{D_B[\boldsymbol{f}(x+s, y+t)]\}\}, \qquad (x,y) \in Z^2\end{aligned} \tag{13.2}$$

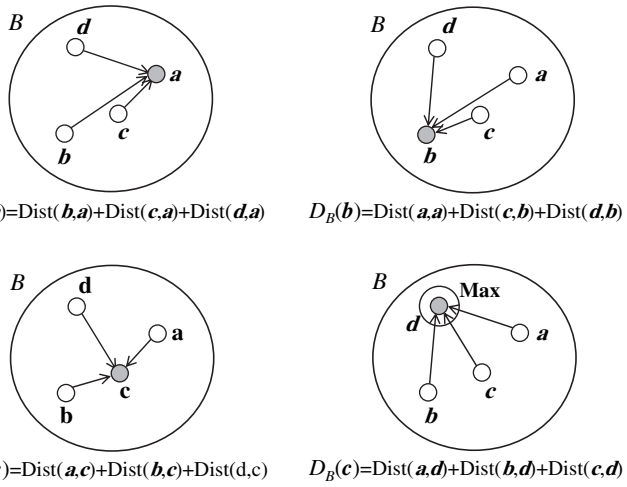

Figure 13.2. Example illustrating the proposed distance-based vector ordering strategy.

where the arg min operator selects the pixel vector which is most highly similar, according to the distance Dist, to all the other pixels in the in the B-neighborhood. On other hand, the flat extended dilation of $\boldsymbol{f}$ by B selects the B-neighborhood pixel vector that produces the maximum value for D_B:

$$\begin{aligned}(\boldsymbol{f} \oplus B)(x,y) &= \{\boldsymbol{f}(x+s', y+t'), (s',t') \\ &= \arg\max_{(s,t)\in Z^2(B)} \{D_B[\boldsymbol{f}(x+s, y+t)]\}\}, \qquad (x,y) \in Z^2 \end{aligned} \tag{13.3}$$

where the arg max operator selects the pixel vector that is most highly different, according to Dist, to all the other pixels in the B-neighborhood. Using the above notation, the multichannel morphological gradient at pixel $\boldsymbol{f}(x,y)$ using B can be simply defined as follows:

$$G_B(\boldsymbol{f}(x,y)) = \text{Dist}((\boldsymbol{f} \oplus B)(x,y), (\boldsymbol{f} \otimes B)(x,y)) \tag{13.4}$$

In this chapter, we assume that Dist is the spectral angle mapper (SAM), a standard metric in hyperspectral analysis. The SAM between two pixel vectors $\boldsymbol{f}(x,y)$ and $\boldsymbol{f}(x',y')$ is given by the following expression: $\text{SAM}[\boldsymbol{f}(x,y), \boldsymbol{f}(x',y')] = \cos^{-1}[\boldsymbol{f}(x,y) \cdot \boldsymbol{f}(x',y') / \| \boldsymbol{f}(x,y) \| \cdot \| \boldsymbol{f}(x',y') \|]$. It should be noted that the proposed extended operators are vector-preserving in the sense that no vector (spectral constituent) that is not present in the input data is generated as a result of the extension process. An important ambiguity not sufficiently explored in previous work has to do with the fact that the ordering imposed above is not injective in general; that is, two or more distinct vectors may output the same minimum or maximum distance. A solution suggested in the literature is to break the tie by using a space-filling

curve such as a Peano curve [5, 6]. However, we believe that the total ordering so created is rather artificial and lacks physical interpretation in remote sensing applications. A further approach is to apply component transformations such as the principal component analysis or maximum noise transform [7] and then consider the first component only [8]. This approach discards significant information that can be very useful for the discrimination of different materials. In this work, we explore the effectiveness of a more physically meaningful approach, based on applying component transformations to bring the data from a high order dimension to a low order dimension, and then exploit all the information available in the reduced feature space to separate the different land cover classes. Our proposed partial solution to alleviate this problem is given by the following tiebreak approach, which is defined in the context of remote sensing applications:

1. Apply a component transformation h to the original N-dimensional data $\boldsymbol{f}$, aimed at increasing the spectral separability between the observed land-cover surfaces in the reduced feature space. Two widely used techniques in remote sensing are considered: principal component analysis (PCA) and maximum noise fraction (MNF). In the two cases, a new M-dimensional data set $\boldsymbol{g} = h(\boldsymbol{f})$ is obtained, with $M \leq N$ and the resulting M channels ordered in terms of decreasing information content. By virtue of the considered transformations, features not discernible in the original data may be evident in the reduced feature space.
2. Order the pixels by according to their spectral properties in the reduced feature space. If we assume that $\boldsymbol{g}(x', y') = [g_1(x', y'), \ldots, g_M(x', y')]^T$ and $\boldsymbol{g}(x'', y'') = [g_1(x'', y''), \ldots, g_M(x'', y'')]^T$ are the correspondent representations of $\boldsymbol{g}(x', y')$ and $\boldsymbol{g}(x'', y'')$ in the M-dimensional space, a partial solution to address multiple suprema or infima in the original N-dimensional space is to order $\boldsymbol{g}(x', y')$ and $\boldsymbol{g}(x'', y'')$ instead of $\boldsymbol{f}(x', y')$ and $\boldsymbol{f}(x'', y'')$, respectively. Following available approaches in the literature [9], we explore two different alternatives to accomplish the above goal:

   (a) *D-ordering*. This approach is based on the application of function D_B to the two M-dimensional representations of the original pixels, that is, $D_B[\boldsymbol{g}(x', y')]$ and $D_B[\boldsymbol{g}(x'', y'')]$. The pixels may now be ordered according to the ordered values of their cumulative distances in the reduced feature space.

   (b) *R-ordering about the centroid*. This approach is based on ordering of the multivariate reduced sample pixels $\boldsymbol{g}(x', y')$ and $\boldsymbol{g}(x'', y'')$ according to their spectral similarity (either SAM or SID) to a preselected location $\boldsymbol{c}_B(x, y)$, that is, the centroid of all the reduced multivariate samples which are located in the spatial neighborhood of $\boldsymbol{g}(x, y)$ defined by B. According to our interpretation in terms of spectral purity, the centroid is the most highly mixed pixel in the B-neighborhood [10]. It can be simply calculated by taking advantage of our multichannel erosion operation as follows: $\boldsymbol{c}_B(x, y) = (\boldsymbol{g} \ominus B)(x, y)$.

To conclude this subsection, we emphasize that the proposed approach represents only a partial solution to the problem since ties in the reduced M-dimensional feature space after the component transformation may still occur. However, we have experimentally tested that using both PCA and MNF in conjunction with the proposed ordering schemes can significantly reduce the percentage of ties found in the original N-dimensional space, as will be demonstrated by experiments in Section 13.5.

13.3. MORPHOLOGICAL PROCESSING OF HYPERSPECTRAL IMAGE SCENES

This section develops two innovative algorithms for processing hyperspectral images using mathematical morphology concepts. The first one makes use of the concept of morphological profile to produce feature vectors for supervised classification using neural networks. The second one is a fully unsupervised approach that expands the morphological waterhsed transformation to hyperspectral analysis. Parallel processing support for these two algorithms is provided in the following section.

13.3.1. Morphological Profile-Based Classification Algorithm

In this subsection, we propose a supervised classification approach that systematically analyzes spatial and spectral patterns in simultaneous fashion. The classifier makes use of sequences of multichannel transformations with SEs of varying width. A similar approach was applied by Pesaresi and Benediktsson [11], who used a composition of mono-channel morphological operations based on SEs of different sizes in order to characterize image structures in high-resolution grayscale urban satellite data. In this work, we use the concept of multichannel morphological profile, defined as a vector where a measure of the spectral variation of the result of pseudo-morphological transformations is stored for every step of an increasing SE series [12]. Following previous work by Benediktsson et al. [13], we use an artificial neural network-based approach for the classification of the resulting morphological features. Below, we describe the proposed algorithm in two stages: (i) morphological feature extraction and (ii) neural network-based classification.

13.3.1.1. Morphological Feature Extraction. The concept of morphological profile relies on opening and closing by reconstruction [2] a special class of morphological transformations that do not introduce discontinuities and which therefore preserve the shapes observed in input images. The basic contrast imposed by conventional opening and closing versus reconstruction-based opening and closing can be described as follows: Conventional opening and closing remove the parts of the objects that are smaller than the SE, whereas opening and closing by reconstruction either completely removes the features or retains them as a whole. In order to define the concept of multichannel morphological profiles using a simple notation, we have adapted the terminology used by Soille [2], where the spatial coordinates of pixel vectors have been ommitted from the formulation for simplicity.

It should be noted, however, that multichannel morphological profiles defined below are calculated for each pixel vector in the input data. First, we define the geodesic dilation operator $\delta_B^{(1)}$ of $\boldsymbol{f}$ under $\boldsymbol{g}$ as

$$\delta_B^{(1)}(\boldsymbol{f},\boldsymbol{g}) = \max\{\delta_B(\boldsymbol{f}),\boldsymbol{g}\} \tag{13.5}$$

where $\boldsymbol{f}$ and $\boldsymbol{g}$ are multichannel images and $\delta_B(\boldsymbol{f})$ is the elementary pseudo-dilation. Similarly, we define the geodesic erosion $\varepsilon_B^{(1)}$ as

$$\varepsilon_B^{(1)}(\boldsymbol{f},\boldsymbol{g}) = \min\{\varepsilon_B(\boldsymbol{f}),\boldsymbol{g}\} \tag{13.6}$$

where $\varepsilon_B(\boldsymbol{f})$ is the elementary erosion. Then, successive geodesic dilations and erosions can respectively be obtained by

$$\begin{aligned}
\delta_B^{(k)}(\boldsymbol{f},\boldsymbol{g}) &= \underbrace{\delta_B^{(1)}\Big(\delta_B^{(1)}\Big(\cdots\delta_B^{(1)}(\boldsymbol{f},\boldsymbol{g}),\boldsymbol{g}\Big),\boldsymbol{g}\Big)}_{k\text{ times}} \quad \text{and} \\
\varepsilon_B^{(k)}(\boldsymbol{f},\boldsymbol{g}) &= \underbrace{\varepsilon_B^{(1)}\Big(\varepsilon_B^{(1)}\Big(\cdots\varepsilon_B^{(1)}(\boldsymbol{f},\boldsymbol{g}),\boldsymbol{g}\Big),\boldsymbol{g}\Big)}_{k\text{ times}}
\end{aligned} \tag{13.7}$$

The reconstruction by dilation of $\boldsymbol{f}$ under $\boldsymbol{g}$ is then given by $\rho_B^{\delta}(\boldsymbol{f},\boldsymbol{g}) = \delta_B^{(\infty)}(\boldsymbol{f},\boldsymbol{g})$, i.e., until idempotence [2]. Similarly, the reconstruction by erosion of $\boldsymbol{f}$ under $\boldsymbol{g}$ is given by $\rho_B^{\varepsilon}(\boldsymbol{f},\boldsymbol{g}) = \varepsilon_B^{(\infty)}(\boldsymbol{f},\boldsymbol{g})$. With the above definitions in mind, the opening by reconstruction of size k of an image $\boldsymbol{f}$ can be simply defined as the reconstruction of $\boldsymbol{f}$ from the erosion of size k of $\boldsymbol{f}$:

$$\gamma_B^{(k)}(\boldsymbol{f}) = \rho_B^{\delta}\Big(\varepsilon_B^{(k)}(\boldsymbol{f}),\boldsymbol{f}\Big) \tag{13.8}$$

and the closing by reconstruction is defined by duality:

$$\phi_B^{(k)}(\boldsymbol{f}) = \rho_B^{\varepsilon}\Big(\delta_B^{(k)}(\boldsymbol{f}),\boldsymbol{f}\Big) \tag{13.9}$$

Using (13.8) and (13.9), multichannel profiles are defined as follows. The opening profile is defined as the vector $\boldsymbol{p}_i^{\gamma}(\boldsymbol{f}) = \Big\{\gamma_B^{(i)}(\boldsymbol{f})\Big\}$, while the closing profile is given by $\boldsymbol{p}_i^{\phi}(\boldsymbol{f}) = \Big\{\phi_B^{(i)}(\boldsymbol{f})\Big\}$, with $i = \{0,1,\ldots,k\}$. Here, $\phi_B^{(i)}(\boldsymbol{f}) = \gamma_B^{(i)}(\boldsymbol{f}) = \boldsymbol{f}$ by the definition of opening and closing by reconstruction [2]. We define the combined derivative profile $\Delta\boldsymbol{p}_i$ as the vector:

$$\begin{aligned}
&\Delta\boldsymbol{p}_i = \Big\{D\text{ist}\Big[\gamma_B^{(i)}(\boldsymbol{f}),\gamma_B^{(i-1)}(\boldsymbol{f})\Big]\Big\} \cup \Big\{D\text{ist}\Big[\phi_B^{(i)}(\boldsymbol{f}),\phi_B^{(i-1)}(\boldsymbol{f})\Big]\Big\}, \\
&\text{with } i = \{1,2,\ldots,\mathrm{k}\}
\end{aligned} \tag{13.10}$$

13.3.1.2. Neural Network-Based Classification. Although the resulting morphological feature vectors can be usually regarded as low-dimensional when

compared to the original hyperspectral signals, much redundancy may still be present in the feature vectors. Therefore, the application of feature extraction techniques to the resulting feature set after applying MM sequences is of great interest in order to select the most relevant features for class discrimination. In this work, we use decision boundary feature extraction (DBFE), introduced by Lee and Landgrebe [14]. DBFE has been demonstrated to be a very powerful approach for extracting all the necessary features for classification using neural networks [15]. The resulting features after DBFE-based feature extraction were then used to train a back-propagation neural-network-based classifier with one hidden layer, where the number of hidden neurons was empirically set to twice the number of input features and information classes. Different training-testing sets were used in experiments, as shown by experimental results in Section 13.4.

13.3.2. Morphological Watershed-Based Classification Algorithm

The morphological watershed transformation [16] consists of a combination of seeded region growing [17, 18] and edge detection. It relies on a marker-controlled approach that considers the image data as imaginary topographic relief. Let us assume that a drop of water falls on such a topographic surface. The drop will flow down along the steepest slope path until it reaches a minimum. The set of points of the surface whose steepest slope path reach a given minimum constitutes the *catchment basin* associated with that minimum, while the watersheds are the zones dividing adjacent catchment basins. Another way of visualizing the watershed concept is by analogy to immersion [2]. Starting from every minimum, the surface is progressively flooded until water coming from two different minima meet. At this point, a watershed line is erected. In order to extend the above algorithm to hyperspectral images, we propose a multichannel watershed-based algorithm with two stages: (i) minima selection and (ii) flooding.

13.3.2.1. Minima selection. The key of an accurate watershed-based segmentation resides in the initialization—that is, the selection of ''markers'' or minima from which the transform is started. Following a recent work [19], we hierarchically order all minima according to their *deepness* and then select only those above a threshold. This concept can be easily explained using the immersion simulation. The deepness of a basin would be the level the water would reach, coming in through the minimum of the basin, before the water would overflow into a neighbor basin—that is, the height from the minimum to the lowest point in the watershed line of the basin. Deepness can be computed using morphological reconstruction applied to the multichannel gradient in Eq. (13.4). Given a ''flat'' SE of minimal size, designed by B, and the multichannel gradient $G_B(\boldsymbol{f})$ of an n-dimensional image $\boldsymbol{f}$, morphological reconstruction by erosion of $G_B(\boldsymbol{f})$ using B can be defined as follows:

$$(G_B(\boldsymbol{f}) \otimes B)^t(x,y) = \bigvee_{k\geq 1} \left[\delta_B^t(G_B(\boldsymbol{f}) \otimes B | G_B(\boldsymbol{f}))\right](x,y) \tag{13.11}$$

where

$$\left[\delta_B^t(G_B(\boldsymbol{f}) \otimes B | G_B(\boldsymbol{f}))\right](x,y) = \left[\overbrace{\delta_B \delta_B \cdots \delta_B}^{t\text{times}}(G_B(\boldsymbol{f}) \otimes B | G_B(\boldsymbol{f}))\right](x,y) \quad (13.12)$$

and

$$[\delta_B(G_B(\boldsymbol{f}) \otimes B | G_B(\boldsymbol{f}))](x,y) = \vee\{(G_B(\boldsymbol{f}) \otimes B)(x,y), G_B(\boldsymbol{f}(x,y))\} \quad (13.13)$$

In the above operation, $G_B(\boldsymbol{f}) \otimes B$ is the standard erosion of the multichannel gradient image, which acts as a ''marker'' image for the reconstruction, while $G_B(\boldsymbol{f})$ acts as a ''mask'' image. Reconstruction transformations always converge after a finite number of iterations, t—that is, until the propagation of the marker image is totally impeded by the mask image. It can be proven that the morphological reconstruction $(G_B(\boldsymbol{f}) \otimes B)^t$ of $G_B(\boldsymbol{f})$ from $G_B(\boldsymbol{f}) \otimes B$ will have a watershed transform in which the regions with deepness lower than a certain value v have been joined to the neighbor region with closer spectral properties; that is, parameter v is a minima selection threshold.

13.3.2.2. Flooding. In this section, we formalize the flooding process following a standard notation [2]. Let the set $P = \{\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_k\}$ denote the set of k minimum pixel vectors selected after multidimensional minima selection. Similarly, let the catchment basin associated with a minimum pixel $\boldsymbol{p}_i$ be denoted by $\mathrm{CB}(\boldsymbol{p}_i)$. The points of this catchment basin which have an altitude less than or equal to a certain deepness score d [19] are denoted by

$$\mathrm{CB}_d(\boldsymbol{p}_i) = \{\boldsymbol{f}(x,y) \in CB(\boldsymbol{p}_i) | \mathrm{Dist}(\boldsymbol{p}_i, \boldsymbol{f}(x,y)) \leq d\} \quad (13.14)$$

We also denote by $\boldsymbol{X}_d = \cup_{i=1}^k CB_d(\boldsymbol{p}_i)$ the subset of all catchment basins which contain a pixel vector with a deepness value less than or equal to d. Finally, the set of points belonging to the regional minima of deepness d are denoted by $\mathrm{RMIN}_d(\boldsymbol{f}(x,y))$. The catchment basins are now progressively created by simulating the flooding process. The first pixel vectors reached by water are the points of highest deepness score. These points belong to $\mathrm{RMIN}_{\boldsymbol{p}_j}(\boldsymbol{f}(x,y)) = \boldsymbol{X}_{D_B(\boldsymbol{p}_j)}$, where $\boldsymbol{p}_j$ is the deepest pixel in P; that is, $D_B(\boldsymbol{p}_j)$ is the minimum, with $1 \leq j \leq k$. From now on, the water either (a) expands the region of the catchment basin already reached by water or (b) starts to flood the catchment basin whose minima have a deepness equal to $D_B(\boldsymbol{p}_l)$, where $\boldsymbol{p}_l$ is the deepest pixel in the set of $P - \{\boldsymbol{p}_j\}$. This operation is repeated until $P = \emptyset$. At each iteration, there are three possible relations between a connected component $\boldsymbol{Y}$ and $\boldsymbol{Y} \cap \boldsymbol{X}_{D_B(\boldsymbol{p}_j)}$:

1. If $\boldsymbol{Y} \cap \boldsymbol{X}_{D_B(\boldsymbol{p}_j)} = \emptyset$, then a new minimum $\boldsymbol{Y}$ has been discovered at level $D_B(\boldsymbol{p}_l)$. In this case, the set of all minima at level $D_B(\boldsymbol{p}_l)$—that is, $\mathrm{RMIN}_{\boldsymbol{p}_l}(\boldsymbol{f}(x,y))$ —will be used for defining $\boldsymbol{X}_{D_B(\boldsymbol{p}_l)}$.

2. If $\boldsymbol{Y} \cap \boldsymbol{X}_{D_B(\boldsymbol{p}_j)} \neq \emptyset$ and is connected, then the flooded region is expanding and $\boldsymbol{Y}$ corresponds to the pixels belonging to the catchment basin associated with the minimum and having a deepness score less than or equal to $D_B(\boldsymbol{p}_l)$, i.e. $\boldsymbol{Y} = CB_{D_B(\boldsymbol{p}_l)}(\boldsymbol{Y} \cap \boldsymbol{X}_{D_B(\boldsymbol{p}_j)})$.
3. Finally, if $\boldsymbol{Y} \cap \boldsymbol{X}_{D_B(\boldsymbol{p}_j)} \neq \emptyset$ and is not connected, then the flooded regions of the catchment basins of two distinct minima at level $D_B(\boldsymbol{p}_j)$ are expanding and merged together.

Once all levels have been flooded, the set of catchment basins of a multidimensional image $\boldsymbol{f}$ is equal to the set $\boldsymbol{X}_{D_B(\boldsymbol{p}_m)}$, where $\boldsymbol{p}_m$ is the least deep pixel in P; that is, $D_B(\boldsymbol{p}_m)$ is the maximum, with $1 \leq m \leq k$. The set of catchment basins following multidimensional watershed can be represented as a set $\{\mathrm{CB}(\boldsymbol{p}_i)\}_{i=1}^{k}$, where each element corresponds to the catchment basin of a regional minimum of the input image $\boldsymbol{f}$. This is the final segmentation output for the algorithm.

13.4. PARALLEL IMPLEMENTATIONS

In this section, we develop parallel implementations for the two mathematical morphology-based algorithms addressed in the previous section. To reduce code redundancy and enhance reusability, our goal was to reuse much of the code for the sequential algorithms in the parallel implementations. The kernel (SE)-based nature of morphological algorithms introduces some border-handling and overlapping issues that are also carefully addressed below. Before describing our parallel implementations, we first discuss data partitioning schemes for hyperspectral data.

13.4.1. Data Partitioning Strategy

Two types of partitioning can be exploited in the considered application: spectral-domain partitioning and spatial-domain partitioning. Spectral-domain partitioning subdivides the data volume into small cells or subvolumes made up of contiguous spectral bands and assigns one or more subvolumes to each processor. It should be noted that most information extraction techniques for hyperspectral imaging focus on analyzing the data based on the properties of spectral signatures; that is, they utilize with the information provided by each pixel vector *as a whole*. With a spectral-domain partitioning model, each pixel vector may be split amongst several processors, and the calculations made for each spectral signature would need to originate from several processors. Although such spectral-domain partitioning strategy has been used in very low-dimensional remote sensing applications (e.g., those based on color images), it was soon discarded in our framework due to several reasons [20]. First, spatial-domain partitioning is a more natural approach for neighbor-based image processing, because many image processing operations require the same function to be applied to a small set of elements around each pixel vector in the input data volume. A second major reason has to do with the overhead introduced by inter-processor communications. Since most hyperspectral algorithms

extract information based on spectral signatures *as a whole*, partitioning the data in the spectral domain would linearly increase communications with the increase in the number of processing elements, thus complicating the design of parallel algorithms. A final issue is code reusability; to reduce code redundancy and enhance portability, it is desirable to reuse much of the sequential code in the design of the parallel version.

13.4.2. Parallel Morphological Profile-Based Classification Algorithm

Multichannel morphological erosion and dilation operations are characterized by their regularity. Therefore, in order to paralellelize the construction of morphological profiles, a desirable goal is to be able to provide parallel support for standard morphological erosion and dilation since these two operations are used as a baseline to construct morphological profiles in iterative fashion. In this subsection, we provide a simple parallel strategy for optimization of standard multichannel erosions and dilations which has been used to improve computational complexity of the proposed morphological profile-based classification algorithm.

To achieve this goal, we first partition the data in the spatial domain (a pixel vector is never partitioned) so that the size of partitions assigned to the different processors is proportional to their speeds [21]. Then, morphological operations (erosion and dilation) are carried out in parallel by the different processing nodes so that each node works independently, making use of its local data portion only, applying the formulation given in Section 13.2.2. An important issue in SE-based morphological image processing operations of this kind is that accesses to pixels outside the spatial domain of the input image are possible. This is particularly so when the SE is centered on a pixel located in the border of the original image. In sequential implementations, it is common practice to redirect such accesses according to a predefined border handling strategy. In our application, a border handling strategy is adopted when the location of the SE is such that some of the pixel positions in the SE are outside the input image domain (see Figure 13.3a).

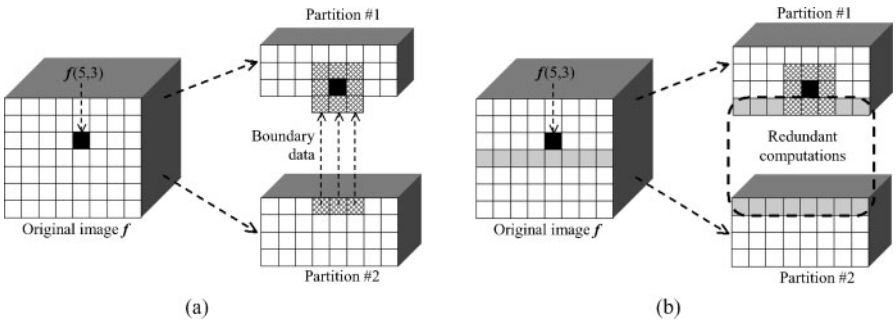

Figure 13.3. (a) 3×3-pixel structuring element computation split among two processing nodes. (b) Introduction of redundant computations to minimize inter-processor communication in a 3×3-pixel structuring element computation.

In this situation, only those pixels inside the image domain are read for the morphological calculation. This strategy is equivalent to the common mirroring technique used in digital image processing applications.

Apart from the border handling strategy above, a function to update overlapping parts of partial data structures has been implemented in order to avoid interprocessor communication when the SE computation is split amongst several different processing nodes (see Figure 13.3b). Here, we use an *overlap mapping* strategy based on replicating the pixel vectors at the border of a partition so that each workstation can perform all the calculations independently, with no need to know which other workstations have pixels close to the boundary of the partition. It should be noted that Figure 13.3b gives a simplified view, because some steps of the operation are not shown. For example, depending on how many adjacent spatial-domain partitions are involved in the parallel computation of a SE, it may be necessary to place a *scratch border* around each partition to completely avoid inter-processor communication. It is also important to note that the amount of redundant information introduced by the overlapping scatter depends on the size of the SE used in the computation. In order to perform the final data partitioning in both cases, we adopt a simple hybrid methodology that consists of two main steps:

1. Partition the hyperspectral data set so that the number of rows in each partition is proportional to the speed of the processor, assuming that no upper bound exists on the number of pixel vectors that can be stored by the processor (at the same time, replicate necessary information for the overlap partitioning strategy). If the number of pixel vectors in each partition assigned to each processor is less than the upper bound on the number of elements that can be stored by the processor, we have an optimal distribution.
2. For each processor, check if the number of pixel vectors assigned to it is greater than the upper bound on the number of elements that it can store. For all the processors whose upper bounds are exceeded, assign them a number of pixels equal to their upper bounds. Now, we solve the partitioning problem of a set with remaining pixel vectors (including those resulting from data replication in the overlap strategy) over the remaining processors. We recursively apply this procedure until all the elements have been assigned.

13.4.3. Parallel Morphological Watershed-Based Classification Algorithm

Parallelization of watershed algorithms that simulate flooding is not a straightforward task. From a computational point of view, these algorithms are representative of the class of irregular and dynamic problems [22]. Moreover, the watershed process has a very volatile behavior, starting with a high degree of parallelism that very rapidly diminishes to a much lower degree of parallelism. Our proposed parallel implementation uses a simple master–slave model. The master processor divides the multichannel image $\boldsymbol{f}$ into a set of subimages $\boldsymbol{f_i}$ which are sent to different processors, so that the domain of each subimage is an extended subdomain given

by D_{fi}^{e}. The slave processors run the classification algorithm on the respective subimages and also exchange data among themselves for uniform segmentation. After the segmented regions become stable, the slaves send the output to the master, which combines all of them in a proper way and provides the final segmentation. If we assume that the parallel system has p processors available, then one of the processors is reserved to act as the master, while each of the remaining $p - 1$ processors create a local queue Q_i with $1 \leq i \leq p - 1$.

The minima selection algorithm is run locally at each processor to obtain a set of minima pixels surrounded by nonminima, which are then used to initialize each queue Q_i. Flooding is then performed locally in each processor as in the serial algorithm. However, due to the image division, flooding is confined only to the local subdomain. Therefore, there may exist parts of the subimage that cannot be reached by flooding since they are contained in other subimages. Our approach to deal with this problem is to first flood locally at every deepness score in the subimage. Once the local flooding is finished, each processor exchanges segmentation labels of pixels in the boundary with appropriate neighboring processors. Subsequently, a processor can receive segmentation labels corresponding to pixels in the extended subdomain. The processor must now ''reflood'' the local subdomain from those pixels, a procedure that may introduce changes in segmentation labels of the local subdomain. Communication and reflooding are again repeated until stabilization (i.e., no more changes occur). When the flood–reflood process is finished, each slave processor sends the final segmentation labels to the master processor, which combines them together and performs region merging to produce final set of segmentation labels.

To conclude this subsection, we emphasize that although the flooding–reflooding scheme above seems complex, we have experimentally tested that this parallelization strategy is more effective than other approaches that exchange segmentation labels without first flooding locally at every deepness score. The proposed parallel framework guarantees that the processors are not tightly synchronized [23]. In addition, the processors execute a similar amount of work at approximately the same time, thus achieving load balance.

13.4. EXPERIMENTAL RESULTS

This section describes the experimental analyses conducted to validate the performance of the proposed morphological hyperspectral processing algorithms. The section is structured as follows. First, we provide a description of the hyperspectral data sets used in experiments. Then, we conduct a thorough quantitative and comparative assessment of the two novel morphological classification algorithms introduced in this chapter, along with a study of the impact of using different vector ordering strategies for the definition of morphological operations. The section concludes with an assessment of the parallel performance of the proposed parallel versions, along with a detailed description of the parallel computing architectures used for evaluation purposes.

13.4.1. Hyperspectral Image Data Set

The data set used in experiments is a hyperspectral scene collected by the NASA/Jet Propulsion Laboratory's Airborne Visible Infra-Red Imaging Spectrometer (AVIRIS) system [24] in 1998 over the Salinas Valley, California. The full scene consists of 512 lines by 217 samples with 224 spectral bands from 0.4 μm to 2.5 μm, nominal spectral resolution of 10 nm, and 16-bit radiometric resolution. It was taken at low altitude with a pixel size of 3.7 m. The data include vegetables, bare soils, and vineyard fields. Figure 13.4a shows the entire scene and a subscene of the data set (called hereinafter Salinas A), outlined by a white rectangle, which comprises 83 × 86 pixels. Figure 13.4b shows available ground-truth regions. As shown in Figure 13.4b, ground-truth is available for about two-thirds of the entire Salinas scene. The data set represents a challenging classification problem due to the early growth stage of most of the crops in the area, which resulted in a lot of mixed pixels (in particular, in the subscene labeled as Salinas A, which comprises lettuce romaine classes at different weeks since planting).

13.4.2. Quantitative Assessment of the Morphological Profile-Based Classifier

In this subsection, we test the performance of the proposed morphological profile-based supervised classification approach. To do so, we extracted a random sample

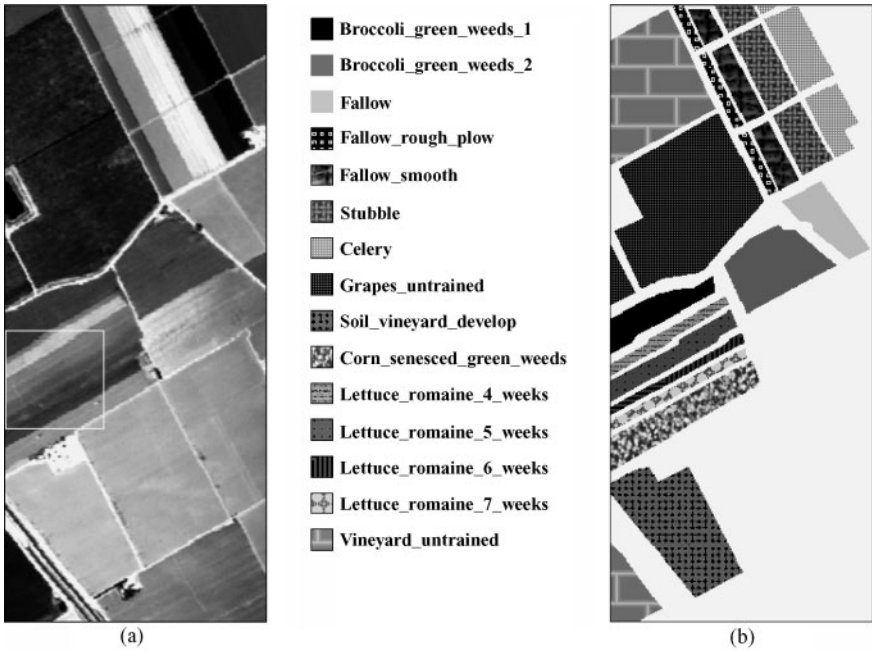

Figure 13.4. (a) Spectral band at 488 nm of an AVIRIS hyperspectral scene comprising agricultural fields in Salinas Valley, California, and subscene (Salinas A) outlined by a white rectangle. (b) Land-cover ground truth classes.

of only 2% of the pixels from the known ground-truth of the 15 ground-truth classes in Figure 13.4b; the number of training pixels selected in each class was proportional to the size of the class. Multichannel morphological profiles were then constructed for the selected training samples, thus allowing for the application of our proposed supervised framework. Finally, we also used the original spectral information contained in the hyperspectral image as an input to our classification system. The resulting features were used to train a back-propagation neural network-based classifier with one hidden layer, where the number of hidden neurons was empirically set to twice the number of input features and information classes. The trained classifier was then applied to the remaining 98% of the known ground pixels in the scene.

Figure 13.5 displays the overall test classification accuracies obtained after applying our classification system to multichannel and mono-channel morphological profiles as a function of the number of opening/closing operations used in the process. Four different approaches were tested in the construction of multichannel morphological operations, given by all possible combinations of the dimensionality reduction transformation (MNF or PCA) plus the reduced ordering strategy (D-ordering or R-ordering). Similarly, two different approaches were considered in the construction of mono-channel profiles based on processing of the first MNF or PCA component. As demonstrated by Figure 13.5, the best overall accuracies were achieved when MNF+D-ordering multichannel morphological profiles were used for feature extraction, followed by PCA+D-ordering. This fact revealed that D-ordering is more appropriate than R-ordering for this application. In all cases, multichannel profiles produced classification results which are clearly better than those found using mono-channel profiles. From Figure 13.5, it is also evident

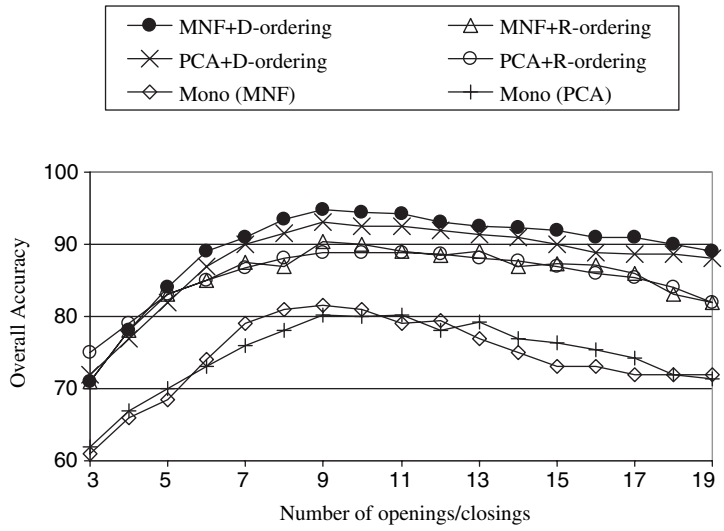

Figure 13.5. Overall classification accuracies for the proposed supervised classifier using different morphological features.

that the width in pixels of patterns of interest in the Salinas AVIRIS scene makes nine opening/closing iterations a reasonable parameter selection for most of the methods tested in this experiment. The construction of morphological feature vectors with larger data dimensions generally causes a loss in the classification performance.

Table 13.1 reports overall (OA), average (AVE), and individual test accuracies for each of the classes in the Salinas data set, using nine iterations. In the table, OA represents the accuracy for all samples whereas AVE represents the average classification accuracy for the samples. The results obtained by using the original spectral information in the hyperspectral scene are also shown for comparison. As can be examined, the OAs exhibited by the D-ordering-based multichannel classifiers are

TABLE 13.1. Overall (OA), Average (AVE), and Individual Test Accuracies in Percentage, Obtained after Applying the Proposed Classification System with Monochannel and Multichannel (MNF- and PCA-based) Profiles with Nine Iterations to the Salinas scene[a]

Class	Original Spectral Information	Mono (MNF)	Mono (PCA)	MNF+D-Ordering	MNF+R-Ordering	PCA+D-Ordering	PCA+R-Ordering
Broccoli_green_weeds_1	78.42	76.21	75.53	82.64	79.36	81.25	79.01
Broccoli_green_weeds_2	80.13	74.58	75.86	86.31	81.26	83.02	81.17
Fallow	92.98	88.51	86.43	98.15	97.54	96.59	95.40
Fallow_rough_plow	96.51	86.77	85.23	96.51	95.30	94.52	92.37
Fallow_smooth	93.72	89.35	86.24	97.63	95.89	95.01	92.89
Stubble	94.71	85.19	86.02	98.96	95.48	98.02	95.17
Celery	89.34	88.40	85.56	98.03	96.75	99.05	93.67
Grapes_untrained	88.02	83.07	82.43	95.34	92.31	93.78	90.67
Soil_vineyard_develop	88.55	78.13	79.63	90.45	87.32	89.13	88.34
Corn_senesced_green_weeds	87.46	70.28	69.83	82.54	80.46	83.90	84.02
Lettuce_romaine_4_weeks	78.86	73.10	72.64	83.21	81.42	82.28	81.49
Lettuce_romaine_5_weeks	91.35	72.57	73.22	82.14	77.43	79.28	78.09
Lettuce_romaine_6_weeks	88.53	74.25	75.39	84.56	80.76	81.81	79.15
Lettuce_romaine_7_weeks	84.85	76.21	77.05	86.57	84.76	84.23	81.47
Vineyard_untrained	87.14	80.04	78.98	92.93	89.23	91.27	87.81
OA	87.25	81.43	80.28	94.82	90.45	93.12	89.03
AVE	87.93	79.73	79.29	90.66	87.96	88.93	87.98

[a]Results using the full spectral information in the original scene are also displayed for comparison.

higher than the OAs provided by R-ordering-based approaches. This confirms the effectiveness of D-ordering with regards to R-ordering in this example. It is also clear from Table 13.1 that the proposed multichannel classifiers outperform single-channel-based approaches in terms of classification accuracies. Most importantly, it should be noted that both D-ordering and R-ordering, when combined with PCA and MNF transformations, produce results that outperform those found using the original spectral information in terms of both OA and AVE. Interestingly enough, however, a deeper analysis of the results reveals some limitations in the proposed techniques. For example, the individual test accuracies exhibited by the MNF+D-ordering, MNF+R-ordering, PCA+D-ordering, and PCA+R-ordering classifiers on the broccoli_green_weeds_1, corn_senesced_green_weeds and four lettuce_romaine (at different weeks since planting) classes are similar to the accuracies produced by the MNF+C-ordering and PCA+C-ordering classifiers on the same classes, and only slightly better than those found by either mono-channel-based morphological classifiers or the original spectral information. It should be noted that the above six classes, all of them contained in the Salinas A subscene (see Figure 13.3a), are dominated by highly mixed pixels such as broccoli plus green weeds, corn senesced plus green weeds, and lettuce romaine plus soil.

13.4.3. Quantitative Assessment of the Morphological Watershed-Based Classifier

In order to test the fully unsupervised classifier, we first used the virtual dimensionality (VD) concept [25] to estimate the number of different classes in the AVIRIS Salinas scene. Here, we masked out the unlabeled pixels in Figure 13.4b in order to assess the algorithm using the full set of ground-truth information available. Interestingly, the VD concept estimated 15 classes which is exactly the number of classes available in Figure 13.4b. It is interesting to note that the VD was able to separate among the four lettuce_romaine classes, which is a significant achievement given the high spectral similarity among those classes. Apart from the number of classes to be extracted, which defined the number of minima ($k = 15$) to be extracted by the minima selection step of the algorithm, parameter v was set automatically using the multilevel Otsu thresholding method as explained in Plaza et al. [26].

Table 13.2 reports the overall (OA), average (AVE), and individual test accuracies for each of the classes in the Salinas data set after applying the proposed multichannel segmentation algorithm using a disk-shaped SE with radius of nine pixels (this size was set empirically after extensive experiments with larger and smaller SEs). For illustrative purposes, results by other two standard unsupervised classification algorithms: The ISODATA [7] and Soille's watershed-based clustering [27] are also reported. As shown by Table 13.2, the use of appropriate SE sizes in the proposed method produced segmentation results that were superior to those found by ISODATA and Soille's watershed-based clustering algorithm. In particular, the best results were obtained when a disk-shaped SE with a radius of nine pixels was used. This is mainly due to the relation between the SE and the spatial properties of

TABLE 13.2. Overall (OA), Average (AVE), and Individual Test Accuracies in Percentage, Obtained After Applying the Proposed Unsupervised Classification System with Mono-channel and Multichannel (MNF- and PCA-based) Profiles with Nine Iterations to the Salinas scene[a]

Class	ISODATA	Soille's	MNF+D-Ordering	MNF+R-Ordering	PCA+D-Ordering	PCA+R-Ordering
Broccoli_green_weeds_1	69.05	70.81	80.45	77.01	78.65	76.23
Broccoli_green_weeds_2	67.01	70.11	84.03	79.44	80.43	78.45
Fallow	82.23	83.28	95.03	94.81	93.25	92.24
Fallow_rough_plow	79.48	80.44	93.28	92.54	91.03	89.00
Fallow_smooth	81.44	81.99	93.99	93.01	92.67	89.34
Stubble	78.00	81.04	95.03	93.24	96.13	92.06
Celery	81.98	80.99	94.28	94.03	96.01	90.45
Grapes_untrained	76.42	77.93	91.23	89.93	91.23	87.79
Soil_vineyard_develop	71.05	76.79	87.34	85.12	87.41	86.19
Corn_senesced_green_weeds	63.92	64.02	80.23	77.76	80.03	82.23
Lettuce_romaine_4_weeks	66.21	68.08	81.34	79.23	79.14	79.44
Lettuce_romaine_5_weeks	65.05	68.72	80.02	75.12	77.81	76.35
Lettuce_romaine_6_weeks	67.57	70.97	81.23	78.21	79.70	76.54
Lettuce_romaine_7_weeks	69.78	72.45	84.16	81.94	82.04	78.23
Vineyard_untrained	73.28	74.25	89.00	87.42	88.76	84.49
OA	74.03	76.17	91.95	89.04	90.16	86.55
AVE	72.24	75.03	88.78	85.12	86.88	83.80

[a]Results using the full spectral information in the original scene are also displayed for comparison.

regions of interest in the scene. Interestingly, as observed in the previous subsection, both D-ordering and R-ordering strategies performed similarly, but MNF-based dimensional reduction proved to be much more effective than PCA-based dimensional reduction. This effect was already observed and thoroughly analyzed in Plaza et al. [28]. Overall, the results shown in Table 13.2 reveal that the proposed algorithm can achieve classification results that are comparable to those reported by the supervised approach in a complex analysis scenario given by agricultural classes with very similar spectral features, in fully unsupervised fashion (complemented with the use of VD to estimate the number of classes). It should also be noted that the two proposed algorithms required more than one hour of computation in a last-generation PC with Intel Centrino 3-GHz processor and 2 GB of RAM, which creates the need for parallel implementations. Performance data for the parallel versions of the two algorithms above are given in the following subsection.

13.4.4. Parallel Performance Evaluation

This subsection provides an assessment of parallel hyperspectral algorithms in providing significant performance gains (without loss of accuracy) with regards to their serial versions. The section is organized as follows. First, we provide an overview of the parallel computing architectures used for evaluation purposes. Second, a

quantitative assessment of the two proposed parallel approaches, in comparison with their respective sequential counterparts, is provided.

The parallel algorithms were coded using the C++ programming language with calls to message passing interface (MPI). They were tested on the Thunderhead Beowulf cluster at NASA's Goddard Space Flight Center (NASA/GSFC). From the early 1990, the overwhelming computational needs of Earth and space scientists have driven NASA/GSFC to be one of the leaders in the application of low cost high-performance computing [29]. In 1997, the HIVE (Highly Parallel Virtual Environment) project was started to build a commodity cluster intended to be exploited by different users in a wide range of scientific applications. The Thunderhead system can be seen as an evolution of the HIVE project (see http://thunderhead.nasa.gov). It is composed of 256 dual 2.4-GHz Intel Xeon nodes, each with 1 Gb of memory and 80 GB of main memory. The total peak performance of the system is 2457.6 Gflops. Along with the 512-processor computer core, Thunderhead has several nodes attached to the core with 2-GHz optical fiber Myrinet. The parallel algorithms tested in this work were run from one of such nodes, called thunder1. The operating system used at the time of experiments was Linux RedHat 8.0, and MPICH was the message-passing library used.

To empirically investigate the scaling properties of the considered parallel algorithms, Figure 13.6 plots the speedup factors (i.e., the ratio of increase in algorithm performance with regards to a single-processor run of the algorithm in a single Thunderhead node), as a function of the number of available processors. Since the use of PCA/MNF or D-ordering/R-ordering seemed to be irrelevant for parallel performance results, Figure 13.6 only displays a general case study for the two considered algorithms. Results in Figure 13.6 reveal that the performance drop from linear speedup in the watershed-based algorithm was more significant than that

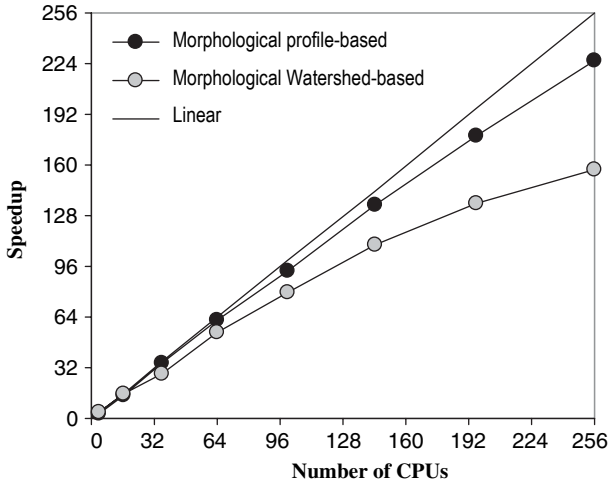

Figure 13.6. Speedup factors achieved by our parallel implementations on Thunderhead.

TABLE 13.3. Load-Balancing Rates and Execution Processing Times for the Parallel Algorithms on Thunderhead.

	Morphological Profile-Based			Watershed-Based		
# CPUs	Time	D_{All}	D_{Minus}	Time	D_{All}	D_{Minus}
1	1874	1.07	1.03	4095	1.35	1.24
4	580	1.05	1.02	1170	1.41	1.30
16	132	1.08	1.04	269	1.42	1.29
36	53	1.07	1.03	144	1.37	1.32
64	30	1.06	1.02	75	1.44	1.31
100	20	1.08	1.04	52	1.45	1.33
144	14	1.07	1.03	37	1.39	1.30
196	11	1.09	1.05	30	1.40	1.28
256	8	1.08	1.03	26	1.45	1.33

observed for the morphological profile-based algorithm as the number of processors increase. This is due to the irregular parallel nature of the flooding stage of the Watershed-based algorithm, which prevents the worker nodes from completing their calculations at exactly the same time. Quite opposite, load balance in the morphological profile-based algorithm was much better due to the regularity in the computations.

In order to fully validate the above remarks, Table 13.3 shows the execution times along with the imbalance scores [30] achieved by the considered parallel algorithms on Thunderhead. The imbalance is defined as $D = R_{\max}/R_{\min}$, where $R_{\max}$ and $R_{\min}$ are the maxima and minima processor run times, respectively. Therefore, perfect balance is achieved when $D = 1$. In the table, we display the imbalance considering all processors, D_{All}, and also considering all processors but the root, D_{Minus}. As we can see from Table 13.3, the parallel version of the morphological profile-based parallel algorithm was able to provide values of D_{All} close to 1 in all cases. Furthermore, the above algorithm provided almost the same results for both D_{All} and D_{Minus} while, for the watershed-based parallel algorithm, load balance was much better when the root processor was not included. Also, the high scores reported for D_{Minus} in this case indicate that the workload is not appropriately balanced among the different processors. The problem of finding an effective workload distribution for region growing algorithms has been identified as a very challenging one in the literature [31], and further work is required to address this issue for the proposed parallel watershed-based algorithm.

For the sake of quantitative comparison, execution times in Table 13.3 reveal that the tested algorithms were able to obtain relevant information from the considered hyperspectral data sets (in light of results in Tables 13.1 and 13.2), but also quickly enough for practical use. For instance, using 256 processors, the morphological profile-based parallel algorithm provided a highly accurate classification result of the Salinas AVIRIS scene in only 8 seconds, while the watershed-based parallel algorithm was able to provide a comparable classification result in 26 seconds, using the same number of processors but in fully unsupervised fashion.

The above results indicate significant improvements over the single-processor runs of the same algorithms, which can take up to more than one hour of computation for the considered problem size, as indicated by Table 13.3. Contrary to common perception that spatial/spectral information extraction algorithms are too computationally demanding for practical use, our results demonstrate that such combined approaches may indeed be very appealing for parallel design and implementation, not only due to the window-based nature of such algorithms but also because they can efficiently distribute the workload among the different processors and reduce sequential computations at the master node, thus enhancing parallel performance.

13.5. CONCLUSIONS AND FUTURE LINES

The recent application of mathematical morphology theory to hyperspectral image data has opened ground-breaking perspectives—in particular, from the viewpoint of naturally integrating the wealth of different sources of information present in the input data. This chapter has described new trends in multichannel processing of hyperspectral imagery by simultaneously considering both spatial and spectral information. A physically meaningful vector organization scheme has been introduced to extend classic morphological operations to high-dimensional spaces, and two different approaches (D-ordering and R-ordering) were used to propose two innovative algorithms for classification of hyperspectral imagery. In the first one, multichannel profiles were used to extract relevant features for classification using neural networks. The second proposed algorithm is a fully unsupervised one which takes advantage of the concept of morphological watershed, originally proposed for grayscale imagery and extended here to the case of multichannel image data. In both cases, component transformations such as PCA or MNF are used to address the issue of partial ordering of pixel vectors. These methods offer a highly representative sample of available and new techniques in morphological hyperspectral analysis research. Our experimental assessment of these two algorithms demonstrated that the first one provides results that are better than those found using both the entire spectral information in the original hyperspectral image and standard, mono-channel morphological processing techniques. The second algorithm provided comparable results (but in fully unsupervised fashion), and slightly better ones than those provided by other similar unsupervised classification approaches. Overall, results in this chapter demonstrated that the combined use of spatial and spectral information in hyperspectral data analysis, achieved via mathematical morphology concepts, can greatly improve the results found by available techniques that consider the spectral information alone—in particular, when adequate vector ordering strategies are employed in the proposed morphological-oriented data processing framework.

A drawback in the proposed approaches has to do with the need to heed a range of filters with increasing SE sizes, a labor that results in a heavy computational burden when processing high-dimensional data. This phenomenon is particularly

relevant for the case of images with large and spectrally homogeneous regions. For that purpose, this chapter has also developed parallel processing support for the two morphological algorithms above. An interesting finding by experiments in this chapter is that spatial/spectral parallel implementations offer a surprisingly simple, yet effective and highly scalable, alternative to standard, spectral-based algorithms for hyperspectral image analysis. Combining the readily available computational power offered by commodity cluster-based parallel architectures such as NASA's Thunderhead system with last-generation sensor and parallel processing technology may introduce substantial changes in the systems currently used by NASA and other agencies for exploiting the sheer volume of Earth and planetary remotely sensed data collected on a daily basis. Although the parallel processing times reported in this chapter are very encouraging, close to (near) real-time processing, we continue our exploration of parallel processing strategies for morphological analysis of hyperspectral image data in real time using hardware-based parallel computing platforms such as field programmable gate arrays (FPGAs). Future work will also include the study of alternative approaches to be used in the extension of morphological operations and an investigation of additional strategies to reduce the impact of partial ordering in the vector space, without losing the physical interpretation inherent to the multichannel remotely sensed data.

ACKNOWLEDGMENTS

The author would like to gratefully thank Professor Chein-I Chang for his guidance, encouragement, and support during a research visit to his laboratory, Remote Sensing Signal and Image Processing Laboratory (RSSIPL). He would also like to thank Drs. John Dorband, Anthony Gualtieri, and James C. Tilton for their collaboration in experiments on the Thunderhead Beowulf cluster. The author would like to acknowledge support received from the Spanish Ministry of Education and Science (Fellowship PR2003-0360), which allowed him to conduct postdoctoral research at University of Maryland Baltimore County and NASA's Goddard Space Flight Center in 2004.

REFERENCES

1. J. Serra, *Image Analysis and Mathematical Morphology*, Academic, New York, 1982.
2. P. Soille, *Morphological Image Analysis: Principles and Applications*, 2nd edition, Springer, Berlin, 2003.
3. S. R. Sternberg, Grayscale morphology, *Computer Graphics and Image Processing*., vol. 35, pp. 333–355, 1986.
4. I. Pitas and C. Kotropoulos, Multichannel L filters based on marginal data ordering, *IEEE Transactions on Signal Processing*, vol. 42, pp. 2581–2595, 1994.
5. J. Chanussot and P. Lambert, Total ordering based on space filling curves for multivalued morphology, in *Mathematical Morphology and Its Applications to Image and Signal*

Processing, edited by H. Heijmans and J. Roerdink, pp. 51–58, Kluwer Academic Publishers, Dordrecht, 1998.

6. C. Regazzoni and A. Teschioni, A new approach to vector median filtering based on space filling curves, *IEEE Transactions on Image Processing*, vol. 6, pp. 1025–1037, 1997.
7. X. Jia, J. A. Richards and D. E. Ricken, *Remote Sensing Digital Image Analysis: An Introduction*, Springer: Berlin, 1999.
8. J. Goutsias, H. Heijmans, and K. Sivakumar, Morphological operators for image sequences, *Computer Vision and Image Understanding*, vol. 62, pp. 326–346, 1995.
9. J. Astola, P. Haavisto, and Y. Neuvo, Vector median filters, *Proceedings of IEEE*, vol. 78, pp. 678–689, 1990.
10. A. Plaza, P. Martinez, R. Perez, and J. Plaza, Spatial/spectral endmember extraction by multidimensional morphological operations, *IEEE Transactions on Geoscience and Remote Sensing*, vol. 40, pp. 2025–2041, 2002.
11. M. Pesaresi and J. A. Benediktsson, A new approach for the morphological segmentation of high resolution satellite imagery, *IEEE Transactions on Geoscience and Remote Sensing*, vol. 39, pp. 309–320, 2001.
12. A. Plaza, P. Martinez, R. Perez, and J. Plaza, A new approach to mixed pixel classification in hyperspectral imagery based on extended morphological profiles, *Pattern Recognition*, vol. 37, pp. 1097–1116, 2004.
13. J. A. Benediktsson, M. Pesaresi, and K. Arnason, Classification and feature extraction for remote sensing images from urban areas based on morphological transformations, *IEEE Transactions on Geoscience and Remote Sensing*, vol. 41, pp. 1940–1949, 2003.
14. C. Lee and D. Landgrebe, Decision boundary feature extraction for neural networks, *IEEE Transactions on Neural Networks*, vol. 8, pp. 75–83, 1997.
15. J. A. Benediktsson, J. A. Palmason, and J. R. Sveinsson, Classification of hyperspectral data from urban areas based on extended morphological profiles, *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, pp. 480–491, 2005.
16. S. Beucher, Watershed, hierarchical segmentation and waterfall algorithm, in *Mathematical Morphology and Its Applications to Image Processing*, edited by E. Dougherty, Kluwer, Boston, 1994.
17. R. Adams and L. Bischof, Seeded region growing, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, pp. 641–647, 1994.
18. A. Mehnert and P. Jackway, An improved seeded region growing algorithm, *Pattern Recognition Letters*, vol. 18, pp. 1065–1071, 1997.
19. N. Malpica, J. E. Ortuño, and A. Santos, A multichannel watershed-based algorithm for supervised texture segmentation, *Pattern Recognition Letters*, vol. 24, pp. 1545–1554, 2003.
20. A. Plaza, D. Valencia, P. Martínez, and J. Plaza, Commodity cluster-based parallel processing of hyperspectral imagery, *Journal of Parallel and Distributed Computing*, vol., pp. 2006.
21. D. Valencia, A. Plaza, P. Martínez, and J. Plaza, On the use of cluster computing architectures to process hyperspectral imagery, in *Proceedings of the IEEE Symposium on Computers and Communications*, Cartagena, Spain, pp. 995–1000, 2005.
22. A. N. Moga and M. Gabbouj, Parallel image component labeling with watershed transformation, *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, pp. 441–450, 1997.

23. A. N. Moga and M. Gabbouj, Parallel marker-based image segmentation with watershed transformation, *Journal of Parallel and Distributed Computing*, vol. 51, pp. 27–45, 1998.
24. R. O. Green et al., Imaging spectroscopy and the airborne visible/infrared imaging spectrometer (AVIRIS), *Remote Sensing of Environment*, vol. 65, pp. 227–248, 1998.
25. C.-I Chang, *Hyperspectral Imaging: Techniques for Spectral Detection and Classification*, Kluwer, New York, 2003.
26. A. Plaza, P. Martinez, R. Perez, and J. Plaza, A quantitative and comparative analysis of endmember extraction algorithms from hyperspectral data, *IEEE Transactions on Geoscience and Remote Sensing*, vol. 42, pp. 650–663, 2004.
27. P. Soille, Morphological partitioning of multispectral images, *Journal of Electronic Imaging*, vol. 5, pp. 252–265, 2001.
28. A. Plaza, P. Martinez, J. Plaza and R.M. Perez, Dimensionality reduction and classification of hyperspectral image data using sequences of extended morphological transformations, *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, no. 3, pp. 466–479, 2005.
29. J. Dorband, J. Palencia, and U. Ranawake, Commodity computing clusters at Goddard Space Flight Center, *Journal of Space Communication*, vol. 1, no. 3, 2003.
30. M. J. Martín, D. E. Singh, J. C. Mouriño, F. F. Rivera, R. Doallo, and J. D. Bruguera, High performance air pollution modeling for a power plan environment, *Parallel Computing*, vol. 29, pp. 1763–1790, 2003.
31. M. G. Montoya, C. Gil, and I. García, The load unbalancing problem for region growing image segmentation algorithms, *Journal of Parallel and Distributed Computing*, vol. 63, pp. 387–395, 2003.