RJ 9754 (84640) March 25, 1994
Computer Science

Research Report

A FAST ALGORITHM FOR MDL-BASED MULTI-BAND IMAGE SEGMENTATION

Tapas Kanungo
Byron Dom
Waybe Niblack
David Steele

IBM Research Division
Almaden Research Center
650 Harry Road
San Jose, CA 95120-6099

IBM **Research Division**
Yorktown Heights, New York ▪ San Jose, California ▪ Zurich, Switzerland

RJ 9754 (84640) March 25, 1994
Computer Science

A FAST ALGORITHM FOR MDL-BASED MULTI-BAND IMAGE SEGMENTATION

Tapas Kanungo*
Byron Dom
Wayne Niblack
David Steele

IBM Research Division
Almaden Research Center
650 Harry Road
San Jose, CA 95120-6099

ABSTRACT:
We consider the problem of image segmentation and describe an algorithm that is based on the Minimum Description Length (MDL) principle, is fast, is applicable to multi-band images, and guarantees closed regions. We construct an objective function that, when minimized, yields a partitioning of the image into regions where the pixel values in each band of each region are described by a polynomial surface plus noise. The polynomial orders and their coefficients are determined by the algorithm. The minimization is difficult because (1) it involves a search over a very large space and (2) there is extensive computation required at each stage of the search. To address the first of these problems we use a region-merging minimization algorithm. To address the second we use an incremental polynomial regression that uses computations from the previous stage to compute results in the current stage, resulting in a significant speed up over the non-incremental technique. The segmentation result obtained is suboptimal in general but of high quality. Results on real images are shown.

*Tapas Kanungo is now at: Intelligent Systems Laboratory; Department of Electrical Engineering, FT-10 University of Washington; Seattle, WA 98195; (tapas@ee.washington.edu)

1. Introduction

1.1. The General Image Segmentation Problem

This paper[1] describes a solution to the problem of unsupervised multiband image segmentation. This is an extension of a previous algorithm[SDNS92]. More precisely, the problem we address is the following. We are given an image in which the ith pixel has associated with it two vectors: $\mathbf{y}_i \in \Re^d$ and $\mathbf{x}_i \in \mathbf{Z}^q$. The components of $\mathbf{x}_i$ are the pixel coordinates in a discrete, q-dimensional space. The components of $\mathbf{y}_i$ represent the measured intensity levels (greylevels) in d different "bands". These may be spectral bands (e.g. "red", "green" and "blue"), different imaging modalities (e.g. an optical image and a range image), "features" (using pattern recognition parlance) computed from the greylevels of the pixels in some neighborhood of pixel i, [2] and so on. We assume that the image represents a real scene consisting of objects, regions, surfaces, etc. Our task is to divide this image into a number of non-overlapping regions whose union is the entire image. The goal is for these regions to correspond to actual regions, objects, surfaces, etc. in the real scene. We want the regions produced by the algorithm to be homogeneous in some sense. There may be a precise measure of homogeneity corresponding to a particular application or we may want the grouping into regions to be similar to one that would be produced by a human given the same task. We consider this to be an important problem that is still open in the sense that a truly general solution has not been found. Our approach here will be to propose a precisely defined problem whose solution is, in many cases of interest, consistent with this more general but somewhat imprecise definition. In doing so, we will make certain assumptions about the images being segmented. In practice these frequently may not hold of course, but, taking a pragmatic view, we will judge the algorithm by how well it does in segmenting images of real scenes.

The solution to the segmentation problem that we present uses the Minimum Description Length Principle (MDL), a natural and powerful tool for regularizing underconstrained optimization problems, to obtain a complexity-based objective function. The originality of our approach is in its combination of generality (multi-band, high-order polynomial surfaces), speed, and this fixed (no adjustable thresholds) MDL-based objective function.

---

[1] This is an expanded version of a conference paper to be presented in June 1994[KDNS94].

[2] Some examples are the local mean, median, maximum, gradient magnitude/direction, Laplacian. More complicated measures are, of course, also possible.

1.2. Related Work

Before beginning a detailed description of our approach, we will briefly survey related work. We will review here only closely related work. For a more complete survey of image segmentation techniques see [HS85, HS92]. The MDL criterion was first used for the problem of image segmentation by Leclerc[Lec89a] where a graylevel image was partitioned into regions, with a two-dimensional polynomial model defined on each region. A continuation minimization procedure[BZ87] led to an algorithm for finding the regions and polynomials, yielding very promising results. This work is the most closely related to ours. There are, however, important differences between his approach and ours: (1) we use a region-merging-based optimization procedure whereas Leclerc uses a continuation scheme. Although the continuation approach is less likely to result in a suboptimal (local minimum) solution than ours, it is much slower and if not allowed to run to convergence, may leave boundary fragments that aren't closed. Our approach never does this; (2) ours treats multiple band images; (3) we explicitly count the cost of the encoding parameters in our MDL formulation; (4) ours is implemented by a fast, incremental computation; (5) Leclerc's algorithm, through its application of the continuation approach has a "relaxation" flavor where pixels determine their new (next iteration) parameter values based on the values of those parameters and the greyvalues of neighboring pixels (and their own) for the current iteration. Our approach is region based, however, always maintaining and merging regions based on their statistics and boundaries. In more recent work[Lec90] Leclerc applied the same approach to region grouping. Our approach lends itself naturally to that problem as well; see [SDNS92].

Other authors have also used MDL for image segmentation. Keeler[Kee90] describes a method in which he segments an image by encoding the topology of the segments (for which he has an efficient encoding), their specific boundaries, and the pixel values in each segment as a noise-corrupted constant grey level. The segmentation is the one for which the encoding length of the topology, boundaries, means, and deviations from means is minimum. Fua and Hanson[FH88] use MDL for a model-based image segmentation. They use geometric constraints on object boundaries (e.g. they are straight lines), allow certain outlying pixels to be excluded to account for shadows, etc., and model only the "objects" in a scene, not the background. Pentland, Darrel and Sclaroff have applied MDL in image and motion segmentation[Pen89, DSP90, DP91]. They use part-based models combined with an optimization algorithm that uses a modified Hopfield-Tank network and a continuation scheme.

MDL has been used to achieve segmentation by simple feature-space clustering. See for example the work of Wallace and Kanade[WK90]. Zhang and Modestino[ZM90] use the AIC (Akaike's information criterion [Aka74]), an information-theoretic criterion that is an alternative to MDL, for image segmentation, also by simple feature-space clustering.

Keren et. al.[KMWP90] apply MDL to the problem of 1D waveform segmentation and experiment with extension of the technique to images by operating on 1D projections of those images.

Besl and Jain [BJ88] also addressed image segmentation using polynomial surface fitting, but the criterion function uses a user-specified threshold for acceptable noise variance and does not account for the model complexity as the MDL principle does. Another approach that uses a similar image model (polynomial surfaces plus Gaussian noise) is applied to 2D images in [LCJ91] and 3D surfaces in [LJ91]. This work is also not based on MDL however, and uses a different optimization algorithm.

1.3. The Problem We Solve

Similar to many of these approaches, to solve this problem we will use the common general approach of formulating an *objective function*, whose global minimum (we assume) corresponds to the best segmentation of the image, then devising an *optimization procedure* that attempts to find this minimum. In formulating this objective function we will assume that the images to be analyzed come from a certain stochastic process, characterized by a family of stochastic models (probability distributions, $p(\mathbf{y}_i)$). The model we assume for this process consists of an ideal partitioning (the segmentation we seek) of the image into regions, $\{\omega_j\}$ (Denote this segmentation by $\Omega = \{\omega_j\}$.) and a separate probability density $p(\mathbf{Y}_j|\beta_j)$ for each region, where $\mathbf{Y}_j$ represents the collection of $\mathbf{y}_i$'s within region j and β_j is a vector of parameters characterizing the distribution. For example, β_j may consist of the mean vector and covariance matrix of a Gaussian distribution or the parameters of a Markov random field. We will use $\boldsymbol{\beta} \triangleq \{\beta_j\}$ to denote the collection of all the parameters for all the regions in Ω.). More specifically, in the work described here we will assume that the pixel values of the regions of the image can be described by polynomial (in spatial coordinates) greyscale surfaces (one per band) to which "white" (spatially uncorrelated) "noise" has been added. We further assume that this noise is Gaussian distributed with (in general) a non-diagonal covariance matrix

i.e. there can be correlation among the bands[3]. For this model we may write:

$$p(\mathbf{Y}_j|\beta_j) \;=\; \prod_{i\in\omega_j} p(\mathbf{y}_i \mid \beta_j), \tag{1.1}$$

where

$$p(\mathbf{y}_i|\beta_j) \;=\; \frac{1}{(2\pi)^{d/2}|\mathbf{\Sigma}_j|^{1/2}} \exp\left\{-\frac{1}{2}[\mathbf{y}_i - \boldsymbol{\mu}_j(\mathbf{x}_i)]^t \mathbf{\Sigma}_j^{-1}[\mathbf{y}_i - \boldsymbol{\mu}_j(\mathbf{x}_i)]\right\}, \tag{1.2}$$

$\boldsymbol{\mu}_j(\mathbf{x})$, the spatially dependent mean of this distribution, is a d-dimensional vector-valued function, whose components are the values of the underlying polynomial surfaces mentioned above; and $\mathbf{\Sigma}_j$ is the covariance matrix for the region ω_j. Note that for this model $\boldsymbol{\beta}$ consists of the polynomial coefficients of the greyscale surfaces and the components of the covariance matrices. Notice also that in this description the region boundaries are composed of the "cracks" between the pixels. In many images, for example those where the optical resolution of the imaging system (lenses, etc.), expressed in units of length, is larger than the pixel size this may seem like an unjustified assumption. Our use of polynomial models allows the existence of "edge" regions in such cases, however.

We have just specified a parametric model for the image. This model has a large number of degrees of freedom that can be adjusted to "fit" (in the sense of statistical estimation) the model to the image. The adjustable parameters consist of the segmentation, $\mathbf{\Omega}$, and the collection of parameters, $\boldsymbol{\beta}$, for the probability densities associated with all the regions comprising $\mathbf{\Omega}$. Thus, viewing this abstractly, a by-product of the process of this model-fitting exercise is a segmentation of the image. That "by-product" is, of course, the end result we seek. The common way to perform such fitting is to specify a goodness-of-fit measure (the most common being mean-square error) which is then minimized by adjusting the various parameters. The values of the so-obtained parameters constitute the best-fit model. This procedure, when expressed in precise statistical terms, is known as *maximum likelihood* (ML) estimation. Specifically ML estimation maximizes the probability, $p(\mathbf{Y}|M)$,[4] where $\mathbf{Y}$ symbolizes the entire collection of $\mathbf{y}$ values for the image and M is a vector variable whose value completely specifies the model.

A problem with performing ML estimation in a case such as this is that there is no bound on the complexity of the model, M, and the more complex it is made,

[3]This inter-band correlation will be especially strong in cases where, for example, the "noise" actually corresponds to material texture in the scene

[4]Actually, doing ML estimation, the goodness-of-fit measure *is* the value of $p(\mathbf{Y}|M)$. In certain cases (e.g. Gaussian models) this is equivalent to minimizing the mean-square error.

the better the fit obtained until the ridiculous limit of every pixel being a separate region is reached. We say that such problems are "ill-posed" or "under-constrained". To correct such problems some way of "regularizing" them must be found. The approach we have chosen to address this problem is to apply the Minimum Description Length Principle (MDL)[Ris78, Ris89]. In this approach the objective function to be minimized is the description length of the data in a suitable "language." We choose MDL for two reasons: (1) It has a strong fundamental grounding, being based on information-theoretic arguments that can be viewed as a formalization of the physicist's *Ockham's razor:* the simplest model explaining the observations is the best [5]; and (2) It results in an objective function with no arbitrary thresholds. To formalize this, the model is used to *encode* (in the sense of data compression) the data in such a way that it can be decoded by a decoder that "knows" only about the model class (the image size, the number of bands and the fact that polynomial Gaussian models will be used)[6]. The model that gives the shortest description length in bits is then chosen as optimum.

There are different ways to reduce this general methodology to an algorithm that can be applied to a given problem (See [Ris89]). The one we use is conceptually straightforward and typically the easiest computationally. It is based on a two-part encoding, where one part consists of an encoding of the model and the other consists of an encoding of the data using the model. Thus the codelength we seek to minimize is:

$$L(\mathbf{Y}, M) = L(M) \; + \; L(\mathbf{Y}|M), \tag{1.3}$$

where, $L(\ldots)$ denotes codelength. This codelength *is* our objective function. In the following section we will derive detailed expressions for the terms in this equation. If the set of possible models were discrete (countable) and we had a prior probability, $P(M)$ on those models, we could let[7] and let $L(\mathbf{Y}|M) = -\log P(\mathbf{Y}|M)$[8]. In this case minimizing equation (1.3) is equivalent to performing Bayesian *maximum a posteriori* (MAP) estimation. If the set of possible models is not countable (the more usual case, which is also the case in this work), however, the situation is more complicated. An expanded discussion of MDL is presented in Appendix A.

[5] Attributed to William of Ockham (1285-1349)

[6] No actual compression or encoding of the image data is performed, but we must outline the process by which it would be encoded in order to derive the expression for the codelength that we require for our objective function.

[7] This $L(M) = -\log P(M)$ connection between codelength's and probabilities comes from one of Shannon's theorems. See [CT91, Abr63] for a discussion.

[8] This connection with conditional probabilities is why we use the conditional notation, $L(\mathbf{Y}|M)$.

2. The Objective Function

As discussed in the introduction, our objective function will be divided into two parts: the codelength of the model, $L(M)$, and the codelength of the data given the model (i.e. encoded using the model), $L(\mathbf{Y}|M)$. In our approach the specification of the model divides naturally into two components, $M = \{\boldsymbol{\Omega}, \boldsymbol{\beta}\}$: the segmentation, $\boldsymbol{\Omega}$, and the distribution parameters, $\boldsymbol{\beta}$. Thus our total code length (equation 1.3) may be written as:

$$L(\mathbf{Y}, \boldsymbol{\Omega}, \boldsymbol{\beta}) = L(\boldsymbol{\Omega}) \; + \; L(\boldsymbol{\beta}|\boldsymbol{\Omega}) \; + \; L(\mathbf{Y}|\boldsymbol{\Omega}, \boldsymbol{\beta}) \tag{2.1}$$

We begin by deriving an expression for $L(\boldsymbol{\Omega})$.

2.1. Encoding Region Boundaries: The Codelength for the Segmentation, $L(\boldsymbol{\Omega})$

We can encode the boundaries by encoding a graph whose nodes represent the boundaries' intersections (either with each other, or with the image frame), and whose edges represent the boundary branches lying between those intersections[9]. To make the boundaries reconstructable from such a graph, we choose one node from each connected component of this graph to be a reference node. To describe a given connected component we start by specifying the location of the reference node, followed by the number of branches from that node, followed by length of the first boundary branch (corresponding to a graph edge), followed by a chain code representing its path along the rectangular grid between the pixels (this chain-code description was also used in [Lec89a].). Thus the description of the entire graph (image partition) has the form:

1. number of connected components
2. description of first connected component
3. description of second connected component
4. and so on ...

[9]The boundary of a region that forms an "island" within a larger region does not have any natural node. We can circumvent this problem by assigning one of its points, say the upper rightmost, to be a node, with a loop edge attached to it.

where the description of each connected component has the form:

1. the location of the reference node, $\mathbf{x}_0$
2. the number of branches from the reference node, ν_0
3. length of the first boundary branch, l_{01}
4. chain code description of first boundary branch, λ_{01}
5. length of the second boundary branch, l_{01}
6. chain code description of second boundary branch, λ_{02}
7. length of the third boundary branch[10] (if applicable), l_{03}
8. chain code description of third boundary branch, λ_{03}
9. the number of branches from the node at the end of branch (01), ν_{01}
10. length of the first boundary branch from node (01), l_{011}
11. chain code description of first boundary branch, λ_{011}
12. and so on ...

Following this for the entire graph would result in duplication because every branch has two ends and can therefore be seen as originating from the nodes at either of these ends. The simple solution to this, however, is to simply not encode a branch that has already been encoded via another path (from the node at its opposite end), in a sense, pretending, at that point, that it doesn't exist.

Each element of the chain-code description of a branch represents the direction of the next step in the chain. Since the number of possible directions is 3, i.e. the number of adjacent grid points (excluding the last visited grid point)[11], the number of bits required for the chain code is $l_i \log 3$. To encode the length of the

[10] It is possible (but not necessary) to choose a reference node with four branches, but we will assume three.

[11] For the branch starting point, the number of possible directions may be less or more than 3, but we ignore this fact.

boundary segment we use Rissanen's "universal prior" for integers[Ris83], which gives the following code length:

$$L^0(l_i) = \log^*(l_i) \ + \log(2.865064) \tag{2.2}$$

where $\log^*(x) = \log x + \log\log x + \log\log\log x \cdots$ up to all positive terms. Thus, associated with arc i, whose length is l_i, is an encoding cost of $L^0(l_i) \ + l_i \log 3$.

When the regions are large, the bulk of the resulting codelength will be the length of description of the branches, so that we can approximate the description length of the boundaries (neglecting the description length of the graph) by $\sum_{\text{all branches}}[l_i \log 3 \ + L^0(l_i)]$, yielding:

$$L(\boldsymbol{\Omega}) \ \approx \ \sum_i (l_i \log 3 + L^0(l_i))$$

Other boundary-encoding schemes are possible, but this one has two main advantages: its regularizing action and its tractability. It is clear that this scheme results in a relatively simple expression for the segmentation codelength and, as will be shown, it is also tractable. The regularizing question is somewhat deeper. Somehow we want our objective function to favor segmentations that are more likely to occur in nature[12]. The scheme we have chosen favors segmentations with shorter total boundary length for a given image size. This means it favors a small number of regions with smooth boundaries. This also seems to be a reasonable measure of complexity, though it does differentiate between some cases where the complexity difference is not clear such as charging a heavier penalty for a large square than for a small one. A more in-depth discussion of this boundary encoding scheme is presented in Appendix B.

2.2. Encoding the Parameters: The Codelength for the Real-Valued Distribution Parameters, $L(\boldsymbol{\beta}|\boldsymbol{\Omega})$

For the coding cost of the real-valued parameters, $\boldsymbol{\beta}$, we use the expression derived by Rissanen in his optimal-precision analysis[Ris83]. For encoding K independent real-valued parameters characterizing a distribution used to describe/encode n data points the codelength he derives is: $(K/2)\log n$. Rissanen derives this expression for the encoding cost of real-valued parameters by optimizing the precision to which

[12]In some cases, of course, one might replace "nature" with a particular application of interest. In this case, however, we seek an objective function and associated algorithm with general applicability.

they are encoded. Encoding them to infinite precision would require an infinite number of bits and there is a trade-off point at which the gain (i.e. decrease) in the codelength of the data due to increasing the precision of the parameters is exactly offset by increased codelength for the parameters. This codelength corresponds to that optimal precision, but is an asymptotic form for large n. During the writing of this paper we have become aware of recent results in this area[Noh93, Ris93]. These results derive better expressions valid for small n. In future work we will utilize these new results.

Applying this result in our case we will have one such term for each region, which results in a total parameter codelength of:

$$L(\boldsymbol{\beta}|\boldsymbol{\Omega}) \;=\; \frac{1}{2}\sum_j K_{\beta_j} \log n_j, \tag{2.3}$$

where K_{β_j} is the number of free parameters in $\boldsymbol{\beta}_j$ and n_j is the number of pixels in region j. For our model we have:

$$K_{\beta_j} \;=\; \frac{d(d+1)}{2} \;+\; dm_j, \tag{2.4}$$

where m_j is the number of polynomial coefficients per band in region j. The first term on the right hand side of equation (2.4) is the number of free parameters in the covariance matrix, $\boldsymbol{\Sigma}_j$. The second term is the number of polynomial coefficients in the spatially varying mean vector, $\boldsymbol{\mu}_j(\mathbf{x})$, which is equal to the number of terms in Θ. For maximum polynomial degree k_j and a two-dimensional ($q = 2$) image $m_j = (k_j + 1)(k_j + 2)/2$, Please see appendix E for the proof and extension to the general case when the image is q-dimensional. Substituting into equation (2.4) yields, for a two dimensional image:

$$K_{\beta_j} \;=\; \frac{d}{2}[(d+1) + (k_j+1)(k_j+2)]. \tag{2.5}$$

It should be mentioned that we have neglected the cost of encoding the polynomial orders $\{k_j\}$. This would add a small, constant number of bits for each region.[13]

[13]If we allow any order, we could use L^0, but, in practice,it won't add more than three bits per region.

2.3. Encoding the Residuals: The Codelength for the Greyvalues within the Regions, $L(\mathbf{Y}|\Omega,\boldsymbol{\beta})$

In this section we will describe the encoding of the residuals (the data given the model) and derive an expression for $L(\mathbf{Y}|M) = L(\mathbf{Y}|\Omega,\boldsymbol{\beta})$. Since our model includes polynomial surfaces fit to the greyvalues in each region, we might think of this step as that of encoding the residuals between the polynomial surface and the actual data. For this reason we will refer to this as the process of "encoding the residuals".

Now let $\mathbf{Y} = [\mathbf{y}_1\mathbf{y}_2\ldots\mathbf{y}_n]^t$ (the total collection of pixel values for the entire image), let $\mathbf{Y}_j$ denote those belonging to the jth region and let n_j the number of pixels in region j. Bear in mind that both n_j and $\mathbf{Y}_j$ are functions of the image partitioning, Ω. To facilitate the notation, however, we will omit the Ω dependence, allowing it to be implicit. Let $p(\mathbf{y}|\beta_j)$ be conditional distribution of a sample $\mathbf{y}$ belonging to the jth region which is characterized by the parameter vector β_j and let the parameter set $\boldsymbol{\beta}\triangleq\{\beta_1,\ldots,\beta_J\}$, where J is the total number of regions in Ω. Then, the conditional distribution $p(\mathbf{Y} \mid \Omega,\boldsymbol{\beta})$ is obtained by forming a product of the individual conditional distributions for all the regions in Ω.

$$p(\mathbf{Y} \mid \Omega,\boldsymbol{\beta}) = \prod_j p(\mathbf{Y}_j \mid \beta_j) \tag{2.6}$$

From Shannon's theorems (see [Abr63]) we know that, when such a distribution is known, the shortest codelength for $\mathbf{Y}$ is given by

$$L(\mathbf{Y}|M) = -\log p(\mathbf{Y}|\Omega,\boldsymbol{\beta}) = \sum_j -\log p(\mathbf{Y}_j \mid \beta_j), \tag{2.7}$$

where the logarithms are base-two.

We now derive an expression for this codelength using the specific assumptions of our model. We use the assumed Gaussian distributions (Equations (1.1) and (1.2)), but in order to use these equations, we need an expression for $\boldsymbol{\mu}_j(\mathbf{x})$, which is a vector-valued function whose components are polynomial greyvalue surfaces of the form:

$$\mu_{jl}(\mathbf{x}) = \sum_{k=1}^{m} \theta_{jlk}\phi_k(\mathbf{x}) \tag{2.8}$$

where μ_{jl} is the l^{th} component of the vector $\boldsymbol{\mu}_j$ and θ_{jlk} is the scalar coefficient for the j^{th} region, the l^{th} band and the kth polynomial basis function. The basis functions $\{\phi_k(\mathbf{x})\}$ are products of various powers of the components of $\mathbf{x}$. (i.e. the two image

spatial coordinates). In matrix form this may be written as: $\boldsymbol{\mu}_j = \boldsymbol{\Phi}_j \boldsymbol{\Theta}_j$ where $\boldsymbol{\mu}_j$ is an $n_j \times d$ matrix of the fitted polynomial surface values (μ values); one for each of the d bands for each of the n_j points in region ω_j. Also, $\boldsymbol{\Phi}_j$ is an $n_j \times m$ matrix of basis function values; one for each of the m basis functions for each of the n_j points. The $n_j \times m$ matrix of regression coefficients is represented by $\boldsymbol{\Theta}_j$. Then, using these definitions, we may rewrite Equation (1.1) obtaining:

$$p(\mathbf{Y}_j \mid \beta_j) = (2\pi)^{-dn_j/2} |\boldsymbol{\Sigma}_j|^{-n_j/2} \exp\left[-\frac{n_j}{2} \text{trace}\left\{\boldsymbol{\Sigma}_j^{-1} S_j\right\}\right] \tag{2.9}$$

where $|\ldots|$ denotes the determinant and S_j is the *sample* covariance matrix defined by: $S_j \triangleq \frac{1}{n_j}(\mathbf{Y}_j - \boldsymbol{\Phi}_j\boldsymbol{\Theta}_j)(\mathbf{Y}_j - \boldsymbol{\Phi}_j\boldsymbol{\Theta}_j)^t$. A proof of this result has been provided in Appendix B. See also [And84].

Using the results presented thus far we can write our objective function as follows.

$$\begin{aligned} L(\mathbf{Y}, \boldsymbol{\Omega}, \boldsymbol{\beta}) = \quad & \textstyle\sum_i (l_i \log 3 + L^0(l_i)) + \sum_j \frac{K_{\beta_j}}{2} \log n_j + \\ & \textstyle\sum_j \frac{n_j}{2} \left[d \log(2\pi) + \log|\boldsymbol{\Sigma}_j| + \text{trace}\left\{\boldsymbol{\Sigma}_j^{-1} S_j\right\}\right], \end{aligned} \tag{2.10}$$

where $\sum_i$ is a sum over all boundary segments and $\sum_j$ is a sum over all regions (not to be confused with the covariance matrix $\boldsymbol{\Sigma}_j$). The three summations correspond to $L(\boldsymbol{\Omega})$, $L(\boldsymbol{\beta}|\boldsymbol{\Omega})$ and $L(\mathbf{Y}|\boldsymbol{\Omega}, \boldsymbol{\beta})$ from left to right in that order.

Since $\mathbf{Y}$ is fixed, we can think of this as an objective function that must be minimized over $\boldsymbol{\Omega}$ and $\boldsymbol{\beta}$. Fortunately, part of this minimization can be performed analytically. In fact, for a given $\boldsymbol{\Omega}$ all the real-valued components (everything except the polynomial orders) of $\boldsymbol{\beta}$ have analytical expressions. For example, for Gaussian distributions the ML estimate (which is also minimum-codelength) for $\boldsymbol{\Sigma}_j$ is $\hat{\boldsymbol{\Sigma}}_j = S_j$. Using this result gives $\text{trace}\left\{\hat{\boldsymbol{\Sigma}}_j^{-1} S_j\right\} = d$. The remaining components of $\boldsymbol{\beta}$ are the polynomial coefficients, $\{\boldsymbol{\Theta}_j\}$, which don't appear explicitly in Equation (2.10), but are required to compute S_j. Expressions for these are derived in the following section.

Further simplifying Equation (2.10) yields the following objective function that can be minimized over all $\boldsymbol{\Omega}$. We use the notation, $\mathcal{L}(\boldsymbol{\Omega})$ (i.e. no functional dependence on $\mathbf{Y}$ and $\boldsymbol{\beta}$) to emphasize the point that during the minimization process, the data, $\mathbf{Y}$, are fixed and the parameters, $\boldsymbol{\beta}$, have analytical expressions in terms of $\mathbf{Y}$ that would appear in an expanded expression for S_j. These are derived in the following section.

$$\mathcal{L}(\boldsymbol{\Omega}) = \frac{n}{2} d(1 + \log 2\pi) + \sum_i [l_i \log 3 + L^0(l_i)] + \frac{1}{2} \sum_j [n_j \log|S_j| + K_{\beta_j} \log n_j] \tag{2.11}$$

Evaluating this expression (Equation (2.11) requires computing the sample covariance matrices, $\{S_j\}$. This can be done as follows (omitting the region subscript j for simplicity of notation).

$$\begin{aligned} n \cdot S &= [\mathbf{Y} - \mathbf{\Phi}\hat{\mathbf{\Theta}}]^t[\mathbf{Y} - \mathbf{\Phi}\hat{\mathbf{\Theta}}] \\ &= \mathbf{Y}^t\mathbf{Y} - \mathbf{Y}^t\mathbf{\Phi}\hat{\mathbf{\Theta}} - \hat{\mathbf{\Theta}}^t\mathbf{\Phi}^t\mathbf{Y} + \hat{\mathbf{\Theta}}^t\mathbf{\Phi}^t\mathbf{\Phi}\hat{\mathbf{\Theta}} \end{aligned} \tag{2.12}$$

But we know that $\mathbf{\Phi}^t\mathbf{\Phi}\hat{\mathbf{\Theta}} = \mathbf{\Phi}^t\mathbf{Y}$. Thus,

$$\begin{aligned} n \cdot S &= \mathbf{Y}^t\mathbf{Y} - \mathbf{Y}^t\mathbf{\Phi}\hat{\mathbf{\Theta}} - \hat{\mathbf{\Theta}}^t\mathbf{\Phi}^t\mathbf{Y} + \hat{\mathbf{\Theta}}^t\mathbf{\Phi}^t\mathbf{Y} \\ &= \mathbf{Y}^t\mathbf{Y} - \mathbf{Y}^t\mathbf{\Phi}\hat{\mathbf{\Theta}} \\ &= \mathbf{Y}^t\mathbf{Y} - \hat{\mathbf{\Theta}}^t[\mathbf{\Phi}^t\mathbf{Y}] \,, \end{aligned} \tag{2.13}$$

since S is symmetric.

To evaluate this expression for S we need an expression for $\hat{\mathbf{\Theta}}$. This is derived in the next section, but as will be discussed in that section, it won't be directly computed for each new region formed during region merging. Rather, incremental formulas will be derived. Using these, $\hat{\mathbf{\Theta}}$ and S for a new combined region will be computed in terms of the results for the two merged regions.

An important feature of this objective function is that it contains no adjustable thresholds or arbitrary parameters of any kind. It is simply derived from "first principles" and applied directly to images. The only choices at our discretion in deriving it were the class of distributions (Gaussian), the functional form (polynomial) of the surfaces and the boundary encoding scheme.

3. The Regression Problem

In the previous section we obtained an expression for our objective function, ($\mathcal{L}(\mathbf{\Omega})$;Equation (2.11)), which we would like to minimize over all $\mathbf{\Omega}$. The sample covariance matrices, $\{S_j\}$ appear in this equation and their calculation involves the problem of fitting multi-variate polynomial functions (surfaces) to discrete multi-variate data (i.e. $\mathbf{Y}$). In fact they (the $\mathbf{S}_j\}$ partially characterize the statistics of the deviations of the data from these surfaces. Before describing the algorithm for finding the best segmentation in the following section, in this section we derive the expressions to be used in calculating S_j.

Here we treat the general problem of fitting multi-variate polynomial functions (surfaces) to discrete multi-variate data of the form $f : \mathbf{Z}^q \rightarrow \mathbf{Z}^d$, where $\mathbf{Z}$ is the set of integers. In the case of one band, 2-D grayscale images, $q = 2$ and $d = 1$.

3.1. Uni-Variate Regression

First consider the case where $f : \mathbf{Z}^q \rightarrow \mathbf{Z}$, i.e., the number of bands $d = 1$. Assume that we are given n ordered data points $(\mathbf{x}_i, y_i)$, $1 \leq i \leq n$ where $\mathbf{x}_i \in \mathbf{Z}^q$ is the spatial coordinate of the i^{th} pixel and $y_i \in \mathbf{Z}$ is its greyvalue. Let $\phi_j(\mathbf{x})$, $1 \leq j \leq m$ be a set of m basis functions such that y_i can be modeled as

$$y_i = \sum_{j=1}^{m} \theta_j \phi_j(\mathbf{x}_i) + \psi_i \tag{3.1}$$

where ψ_i is zero-mean Gaussian noise with variance σ^2, and θ_j are scalar coefficients. The basis functions $\phi_j(\mathbf{x})$ in our case are products of various powers of components of $\mathbf{x}$. e.g. if $\mathbf{x}_1 = (x_{11} \ \cdots \ x_{1q})^t \in \mathbf{Z}^q$, then some examples of $\phi(\mathbf{x}_1)$ are 1, x_{11}, $(x_{11})^3$, $(x_{11})^2(x_{1q})^3$, etc. Using this model we can write an an expression for all the data as follows:

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} \phi_1(\mathbf{x}_1) & \phi_2(\mathbf{x}_1) & \cdots & \phi_m(\mathbf{x}_1) \\ \phi_1(\mathbf{x}_2) & \phi_2(\mathbf{x}_2) & \cdots & \phi_m(\mathbf{x}_2) \\ \vdots & \vdots & \vdots & \vdots \\ \phi_1(\mathbf{x}_n) & \phi_2(\mathbf{x}_n) & \cdots & \phi_m(\mathbf{x}_n) \end{bmatrix} \cdot \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_m \end{bmatrix} + \begin{bmatrix} \psi_1 \\ \psi_2 \\ \vdots \\ \psi_n \end{bmatrix} \tag{3.2}$$

We represent the above equation in matrix form as

$$\mathbf{Y} = \mathbf{\Phi} \cdot \boldsymbol{\theta} + \boldsymbol{\psi} \tag{3.3}$$

where $\mathbf{Y}$ and $\boldsymbol{\psi}$ are $n \times 1$ vectors, $\boldsymbol{\theta}$ is an $m \times 1$ vector and $\mathbf{\Phi}$ is an $n \times m$ matrix. Here $\boldsymbol{\psi}$ is assumed to be zero-mean Gaussian-distributed as $N(0, \sigma^2 I)$, where I is the $n \times n$ identity matrix (i.e. we assume the samples to be uncorrelated).

The regression problem, then, is to find the $\hat{\boldsymbol{\theta}}$ that minimizes

$$\epsilon^2 = ||\mathbf{Y} - \mathbf{\Phi}\boldsymbol{\theta}||^2 \ . \tag{3.4}$$

The solution to this minimization problem is

$$\hat{\boldsymbol{\theta}} = [\mathbf{\Phi}^t\mathbf{\Phi}]^{-1}\mathbf{\Phi}^t\mathbf{Y} \ . \tag{3.5}$$

Although the above equation is correct, it is sensitive numerically, because computing the inverse of $[\mathbf{\Phi}^t\mathbf{\Phi}]$ can be unstable[Gv83, PTaBPF92]. It is better to solve the system:

$$\mathbf{\Phi}^t\mathbf{y} = \mathbf{\Phi}^t\mathbf{\Phi}\boldsymbol{\theta} \ . \tag{3.6}$$

The solution $\hat{\boldsymbol{\theta}}$ for the system of equations $\mathbf{\Phi}^t\mathbf{Y} = \mathbf{\Phi}^t\mathbf{\Phi}\boldsymbol{\theta}$ is the same as the $\hat{\boldsymbol{\theta}}$ that minimizes $||\mathbf{Y} - \mathbf{\Phi}\boldsymbol{\theta}||^2$. (see Appendix A for a proof). Asymptotically, both the techniques are of the same order of complexity but for smaller systems of equations, a considerable amount of computation time can be saved.

3.2. Multi-Variate Regression

Now consider the case when $f : \mathbf{Z}^q \rightarrow \mathbf{Z}^d$, i.e., the number of bands d, is greater than one. In this case we can represent the regression problem as

$$[\mathbf{y}_1\mathbf{y}_2 \cdots \mathbf{y}_n]^t = \mathbf{\Phi} \cdot [\boldsymbol{\theta}_1\boldsymbol{\theta}_2 \cdots \boldsymbol{\theta}_d] + [\boldsymbol{\psi}_1\boldsymbol{\psi}_2 \cdots \boldsymbol{\psi}_d] \ , \tag{3.7}$$

where the $\mathbf{y}_i$ are $d \times 1$ vectors representing the gray values in the d bands at the i^{th} pixel, $\boldsymbol{\theta}_i$ are $m \times 1$ vector of regression coefficients for the i^{th} band, and $\boldsymbol{\psi}_i$ are $n \times 1$ vector of Gaussian noise values in the i^{th} band and distributed as $N(0, \sigma^2 I)$, (I is an $n \times n$ identity matrix[14] and $\mathbf{\Phi}$ is a $n \times m$ matrix. We can write the above equation in a more compact form as

$$\mathbf{Y} = \mathbf{\Phi} \cdot \mathbf{\Theta} + \mathbf{\Psi} \tag{3.8}$$

where $\mathbf{Y}$ and $\mathbf{\Psi}$ are $n \times d$ matrices, $\mathbf{\Theta}$ is a $m \times d$ matrix and $\mathbf{\Phi}$ is a $n \times m$ matrix.

The multi-variate regression problem, then, is to find the $\hat{\mathbf{\Theta}}$ that minimizes the sum of squared residuals

$$\epsilon^2 = \text{trace}\left\{(\mathbf{Y} - \mathbf{\Phi} \cdot \mathbf{\Theta})^t(\mathbf{Y} - \mathbf{\Phi} \cdot \mathbf{\Theta})\right\} \ . \tag{3.9}$$

The above operator in this case is essentially the sum of squares of all entries of the error matrix $\boldsymbol{\epsilon} = (\mathbf{Y} - \mathbf{\Phi} \cdot \mathbf{\Theta})$. The solution to the minimization problem is

$$\hat{\mathbf{\Theta}} = [\mathbf{\Phi}^t\mathbf{\Phi}]^{-1}\mathbf{\Phi}^t\mathbf{Y} \ , \tag{3.10}$$

and the same solution is also obtained by solving the system of equations

$$\mathbf{\Phi}^t\mathbf{Y} = \mathbf{\Phi}^t\mathbf{\Phi}\mathbf{\Theta} \ . \tag{3.11}$$

Substituting the expression for $\hat{\mathbf{\Theta}}$ (Equation (3.10) into our expression for S (Equation (2.13)) yields:

$$S \ = \ \mathbf{Y}^t\mathbf{Y} - \{[\mathbf{\Phi}^t\mathbf{\Phi}]^{-1}\mathbf{\Phi}^t\mathbf{Y}\}^t[\mathbf{\Phi}^t\mathbf{Y}].$$

For numerical reasons this expression will not be evaluated directly, but rather $\hat{\mathbf{\Theta}}$ will be calculated and then substituted into Equation (2.13).

[14]Some confusion is possible here. We are saying that the noise samples in the same band are uncorrelated with each other (i.e. no pixel-to-pixel correlation). This is the "white-noise" assumption. On the other hand the various bands may be correlated with each other when measured for the same pixel. We have assumed that the distribution that characterizes that relationship is $N(\mu(\mathbf{x}), \mathbf{\Sigma})$, where $\mathbf{\Sigma}$ is $d \times d$.

3.3. The Incremental Regression Problem

Because we seek only the image segmentation $\mathbf{\Omega}$, we wouldn't need to compute the regression coefficients, $\mathbf{\Theta}$ (They don't appear explicitly in Equation (4.1).) if it weren't for the fact that they are required to compute the covariance matrix estimates $\{\hat{\mathbf{\Sigma}}_j\}$. For this reason, anything that can be done to improve the efficiency of their calculation and that of the $\{\hat{\mathbf{\Sigma}}_j\}$ will be valuable. In this section we derive *incremental* formulas for computing these polynomial regression coefficients and the covariance matrix of a new merged region from those of the two individual regions merged without having to perform an explicit regression on the data of the merged region.

Consider the following two independent multi-variate regressions:

$$\mathbf{Y}_1 = \mathbf{\Phi}_1 \cdot \mathbf{\Theta}_1 + \mathbf{\Psi}_1 \tag{3.12}$$

$$\mathbf{Y}_2 = \mathbf{\Phi}_2 \cdot \mathbf{\Theta}_2 + \mathbf{\Psi}_2 \,, \tag{3.13}$$

where $\mathbf{Y}_i$ is a $n_i \times d$ data vector, $\mathbf{\Phi}_i$ is a $n_i \times m$ regression matrix, $\mathbf{\Psi}_i$ is a $n_i \times d$ noise matrix, $\mathbf{\Theta}$ is a $m \times d$ regression coefficient matrix.

Assume that the optimal $\hat{\mathbf{\Theta}}_1$ and $\hat{\mathbf{\Theta}}_2$ have already been computed. Now consider the following "concatenated" problem:

$$\begin{bmatrix} \mathbf{Y}_1 \\ \mathbf{Y}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{\Phi}_1 \\ \mathbf{\Phi}_2 \end{bmatrix} \cdot \mathbf{\Theta} + \mathbf{\Psi} \,. \tag{3.14}$$

Let $\mathbf{Y} = [\mathbf{Y}_1^t \quad \mathbf{Y}_2^t]^t$ and let $\mathbf{\Phi} = [\mathbf{\Phi}_1^t \quad \mathbf{\Phi}_2^t]^t$. Then the above equation can be written as

$$\mathbf{Y} = \mathbf{\Phi}\mathbf{\Theta} + \mathbf{\Psi} \,. \tag{3.15}$$

The problem: find a computationally efficient method for computing $\hat{\mathbf{\Theta}}$ and $(n_1 + n_2)S = [\mathbf{Y} - \mathbf{\Phi}\hat{\mathbf{\Theta}}]^t[\mathbf{Y} - \mathbf{\Phi}\hat{\mathbf{\Theta}}]$ for the concatenated system.

3.3.1. Incremental Computation of $\hat{\mathbf{\Theta}}$. As discussed earlier, matrix inverse computations can be unstable and it is better to compute $\hat{\mathbf{\Theta}}$ by solving the linear system of equations

$$\mathbf{\Phi}^t\mathbf{Y} = \mathbf{\Phi}^t\mathbf{\Phi}\mathbf{\Theta} \,. \tag{3.16}$$

Expanding $\mathbf{\Phi}$ and $\mathbf{Y}$ we get

$$\mathbf{\Phi}^t\mathbf{Y} = [\mathbf{\Phi}_1^t \; \mathbf{\Phi}_2^t] \cdot \begin{bmatrix} \mathbf{Y}_1 \\ \mathbf{Y}_2 \end{bmatrix} = [\mathbf{\Phi}_1^t\mathbf{Y}_1 + \mathbf{\Phi}_2^t\mathbf{Y}_2] \tag{3.17}$$

and,

$$\mathbf{\Phi}^t\mathbf{\Phi} \;=\; [\mathbf{\Phi}_1^t\ \mathbf{\Phi}_2^t]\cdot\begin{bmatrix}\mathbf{\Phi}_1\\ \mathbf{\Phi}_2\end{bmatrix}\cdot = \left[\mathbf{\Phi}_1^t\mathbf{\Phi}_1 + \mathbf{\Phi}_2^t\mathbf{\Phi}_2\right]\ . \tag{3.18}$$

Notice that the matrix products $\mathbf{\Phi}_i^t\mathbf{Y}_i$ and $\mathbf{\Phi}_i^t\mathbf{\Phi}_i$ are available since they must have been computed for the individual systems. Moreover, although the matrices $\mathbf{\Phi}$ and $\mathbf{Y}$ are of varying dimensions, the matrix products $\mathbf{\Phi}^t\mathbf{Y}$ and $\mathbf{\Phi}^t\mathbf{\Phi}$ are of always of constant small (relative to the number of pixels in most regions) dimensions, independent of the dimensions of $\mathbf{\Phi}$ and $\mathbf{Y}$, which change with the number of pixels in a region. That is, the matrix product $\mathbf{\Phi}^t\mathbf{Y}$ is $m \times d$ and $\mathbf{\Phi}^t\mathbf{\Phi}$ is $m \times m$. The elements of these matrices are given by:

$$(\mathbf{\Phi}^t\mathbf{Y})_{ij} \;=\; \sum_{l=1}^{n}\phi_i(\mathbf{x}_l)y_{jl} \tag{3.19}$$

$$(\mathbf{\Phi}^t\mathbf{\Phi})_{ij} \;=\; \sum_{l=1}^{n}\phi_i(\mathbf{x}_l)\phi_j(\mathbf{x}_l), \tag{3.20}$$

where y_{jl} is the value of the j^{th} band for the l^{th} pixel.

3.3.2. Incremental Computation of S. In this section we expressions for incrementally computing the covariance matrices, $\{S_j\}$. Rrom Equation (2.13) we have:

$$nS \;=\; \mathbf{Y}^t\mathbf{Y} - \hat{\mathbf{\Theta}}^t[\mathbf{\Phi}^t\mathbf{Y}]. \tag{3.21}$$

In the incremental computation we utilize the fact that $\mathbf{Y}^t\mathbf{Y} = \mathbf{Y}_1^t\mathbf{Y}_1 + \mathbf{Y}_2^t\mathbf{Y}_2$, and $\mathbf{\Phi}^t\mathbf{Y} = [\mathbf{\Phi}_1^t\mathbf{Y}_1 + \mathbf{\Phi}_2^t\mathbf{Y}_2]$, which reduces the number of computations in the incremental computation of the covariance matrix. Furthermore, all the matrices involved in the computation of S ($\hat{\mathbf{\Sigma}}$) and $\mathbf{\Theta}$ have fixed dimensions and therefore the book-keeping involved with dynamically changing region sizes is reduced.

4. Segmentation Algorithm

The problem our algorithm must solve is one of finding the minimum of Equation (2.11) over all $\mathbf{\Omega}$. Obtaining the absolute (global) minimum is infeasible because the search space is so large. For this reason we use a hierarchical algorithm similar to that used in [SDNS92, BG89] to find a good, though perhaps local, minimum. It starts with an initial segmentation of the image. This may be just the image itself, with each pixel considered to be a separate region, or it may consist of larger regions produced by some heuristic device. Starting with this initial segmentation,

the algorithm successively merges pairs of neighboring regions provided that the mergers decrease the total code-length. At each step the pair of regions producing the greatest codelength decrease are merged.

The MDL codelength decrease, δ_{tv}, due to a merger between two neighboring regions, ω_t and ω_v, can be deduced from Equation (2.11):

$$\begin{aligned} \delta_{tv} &= [l_{tv} \log 3 + \log l_{tv}] \\ &\quad + \tfrac{1}{2} [n_t \log |S_t| + n_v \log |S_v| - (n_t + n_v) \log |S_{tv}|] \\ &\quad + \tfrac{1}{2} [K_{\beta_t} \log n_t + K_{\beta_v} \log n_v - K_{\beta_{tv}} \log(n_t + n_v)] \end{aligned} \tag{4.1}$$

where S_{tv} denotes the sample covariance matrix of the combined region $\omega_t \cup \omega_v$. As mentioned above, at each step in the algorithm we search for the two regions ω_t,ω_v that yield the greatest code-length decrease δ_{tv} when merged. The first term expresses the savings due to the fact that a boundary branch drops when the merger occurs. The second term is the increase in the codelength of the actual data values themselves. This results from going to a single distribution from a separate distribution for each region. The third term is the savings associated with the fact that we have fewer model parameters to describe after the merger. K_{β_t}, K_{β_v} and $K_{\beta_{tv}}$ represent the number of parameters in the models representing the regions t, v, and tv, respectively.

The total number of merger steps needed to reach the final classification equals the number of initial regions r_0 minus the final number of regions R (usually $R \ll r_0$). The regions are ordered with a *heap-based priority queue* [Sed90] to select the best merger and at each step a time proportional to $\log r_0$ is required to maintain the queue, thus making the run time proportional to $r_0 \log r_0$. The memory size required by the algorithm equals the total number of regions (both the initial and the newly created, summing up, in the worst case, to $2r_0$) multiplied by the memory size required by the data set of a single region, which is roughly proportional to the average number of neighbors of a single region. This last number is usually much smaller than r_0, and therefore the memory requirements of this hierarchical algorithm are also proportional to r_0, being modest when compared to conventional hierarchical clustering procedures that try to merge every possible pair regardless of spatial location, and therefore requiring a memory size proportional to $\frac{1}{2} r_0(r_0 - 1)$. Moreover, some of the items of the data sets of inactive regions may be erased to save memory space.

The algorithm is run by first fixing the maximum degree of regression polynomials to 0. That is, region greyvalues are represented by piece-wise constant functions.

After the algorithm converges to a segmentation (because it is more expensive to encode the image if any further merging is done), merging is attempted with first order polynomials representing the merged regions. This is continued until the merging converges. The process of incrementing the regression polynomial order and merging until convergence is continued until no merging is accomplished for a particular degree of the polynomial. Note that this process can be stopped at any degree of fit and will still result in closed region boundaries.

5. Experimental Results

To test the algorithm, we implemented it in C on an IBM RISC-System/6000 model 970 and ran it on both synthetic and real images. The synthetic images that were used fit the model assumptions of piece-wise constant, linear and quadratic regions with Gaussian noise. Here we show results on two real images. The segmentation results for various maximum polynomial order fits are shown. The computation time for images of size 128×128 was on the order of 180 seconds.

Our first real image test case (Figure 1) was a real 128×128 two-band (red and blue) image of a small fragment of an electronic circuit. Figures 2, 3, and 4 are the segmentation result when the maximum degree of polynomial allowed are zero, one, and two, respectively. As can be seen in the figure, the result for zero-order polynomials (i.e. piece-wise constant) is quite good, but contains many small regions in the boundary areas. This is due partially to the inadequacy of the piece-wise-constant model in these areas. Many of these fragments get merged when the maximum degree of the polynomial allowed is increased to linear, and then quadratic.

Our second real image test case was the real 128×128 pixel image of a house in Figure 5. This is a single-band grayscale image. The results are quite good for the piece-wise constant case in Figure 6, but there are certain region boundaries that appear inappropriate. This is a kind of contouring effect that is, again, an artifact of the piecewise-constant assumption/model. This problem is partially solved when the maximum degree is allowed to go to piecewise quadratic as shown in Figure 7.

Note that when maximum allowed degree of fit is two (quadratic surfaces), some of the regions still can have piece-wise constant and linear surfaces since the MDL criterion might find it cheaper to encode those regions that way. This model selection is, of course, done automatically using the MDL criterion – there are no heuristic thresholds.

It is interesting to note that a very high percentage of time is spent in merging

neighbor lists (37%). An efficient implementation of merge procedure could speed up the software further. Other sections of the algorithm that take considerable amounts of time are: computing logarithms (10%), and computation of the code-length change, δ_{tv} (15%).

6. Discussion and Conclusions

We have developed an MDL-based objective function for multi-band image segmentation and an efficient segmentation algorithm that performs a sub-optimal minimization of this criterion. The algorithm is incremental and makes use of computations performed in previous stages. The algorithm was tested on both synthetic and real images. The speed and performance of the algorithm on the test images were quite good and no manually adjusted thresholds were required. It should be mentioned that this algorithm can be used to treat texture-based segmentation by using the appropriate texture operators to compute the input bands for this algorithm. An alternate approach to texture will be mentioned below. Natural extensions of this algorithm/work include:

- alternate coding schemes for segment boundaries. For example polygonal coding will be more efficient for images that have polygonal shapes (e.g. aerial images of buildings), and Fourier descriptors for images which have shapes with smooth contours (e.g. images of organs).

- a way of incorporating prior information, (e.g. vertical lines may be more probable than horizontal, or, the probability of 90 degree angle between lines might be higher than other angles), into the MDL objective function.

- other (than polynomial Gaussian) stochastic models: For example, Markov random field models could be used for encoding textures. This would then simultaneously segment images into textured and piece-wise smooth regions.

- extend the boundary coding scheme to allow coding of three-dimensional surfaces. This would help in segmenting three-dimensional objects in multi-dimensional (volume) data, e.g., CT images of the heart or other organs.

- a more rigorous analysis of small-sample-size effects on the covariance matrix estimate (and therefore the objective function) when the region sizes are small in the initial states of clustering.

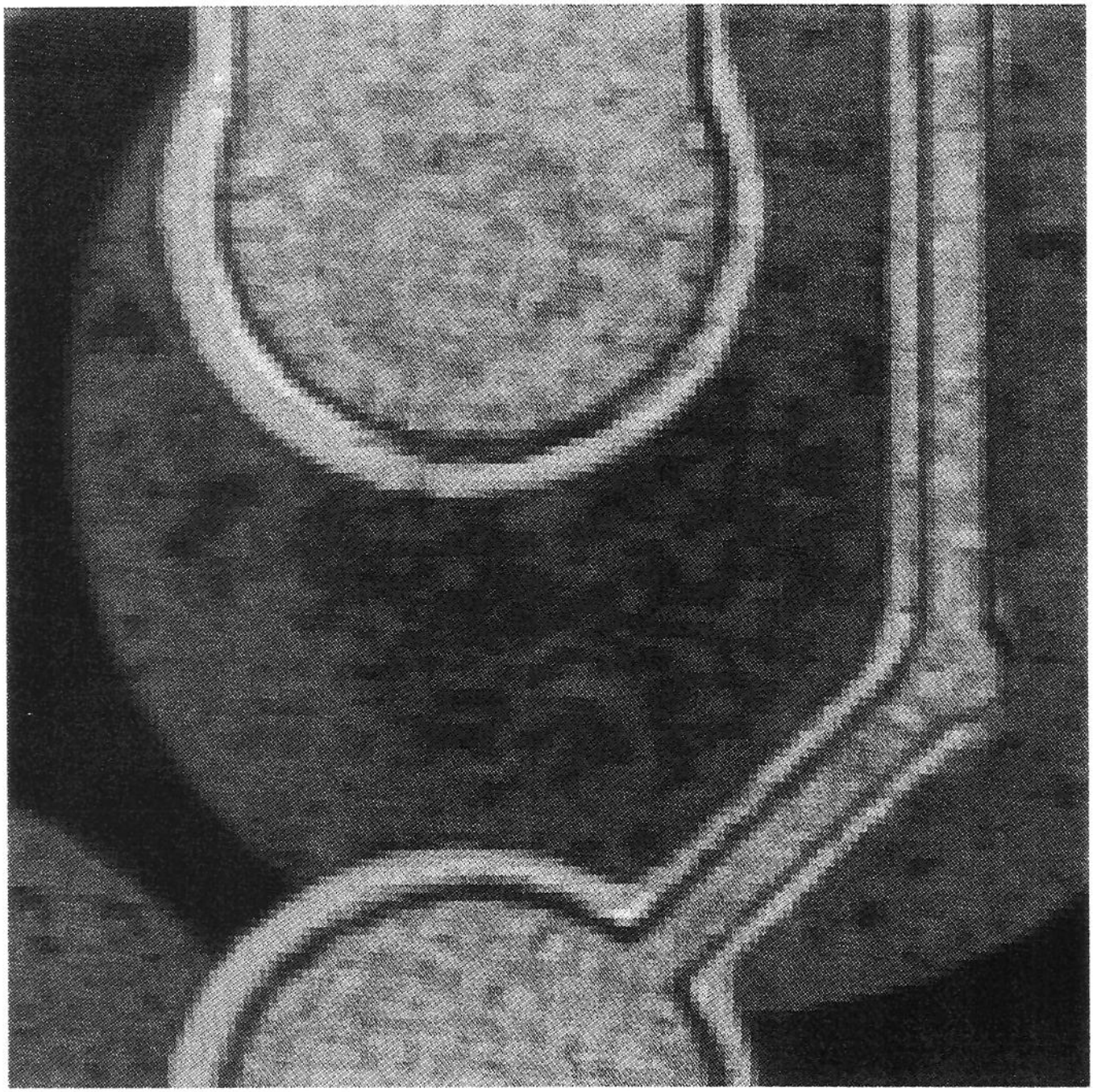

Figure 1: The grayscale image of the red band of a real, two-band (red and blue), image of a small fragment of an electronic circuit.

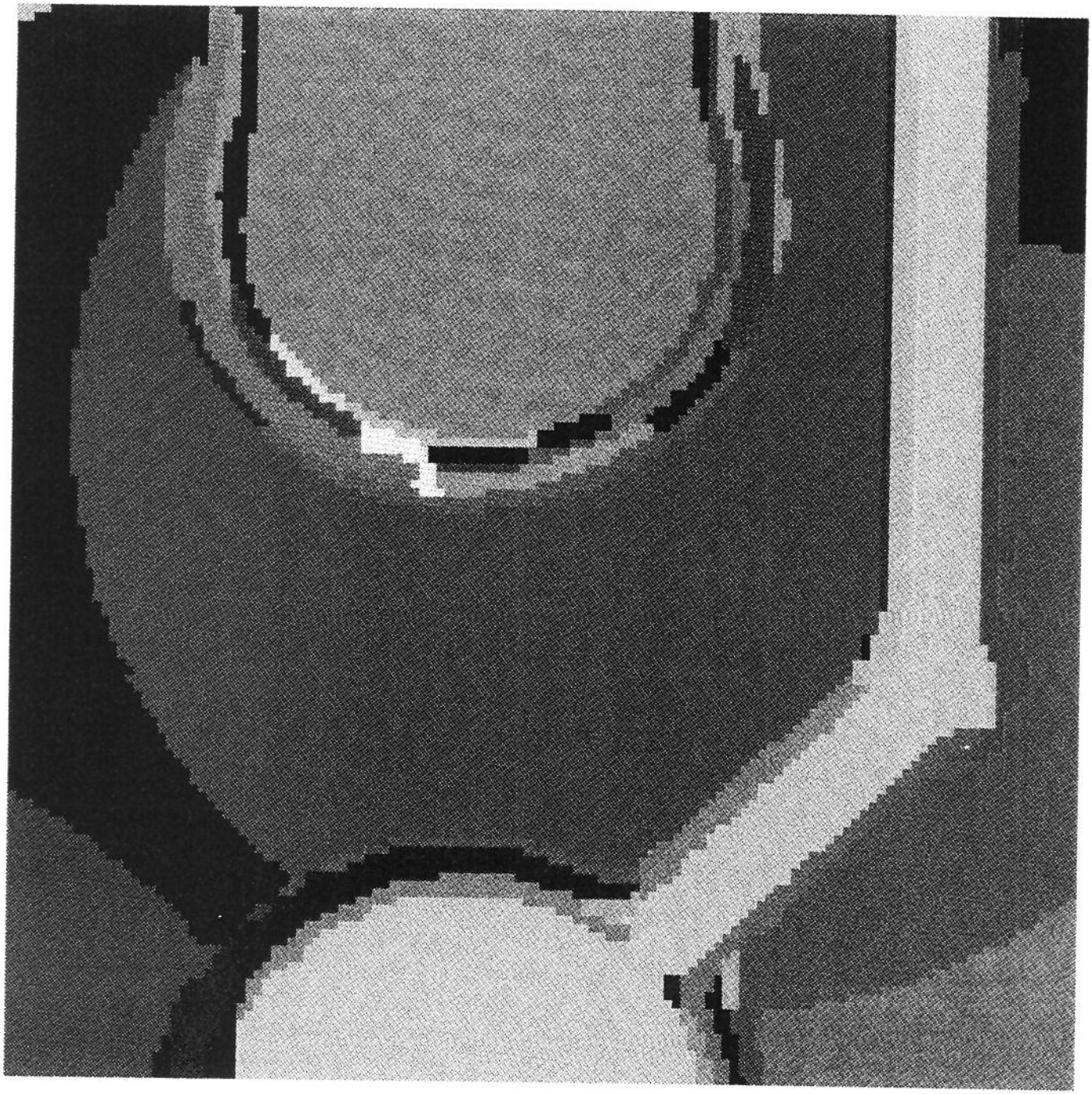

Figure 2: Segmentation result for the electronic circuit image. In this run, the maximum degree of polynomial was zero. That is, a piece-wise constant model was used for each region.

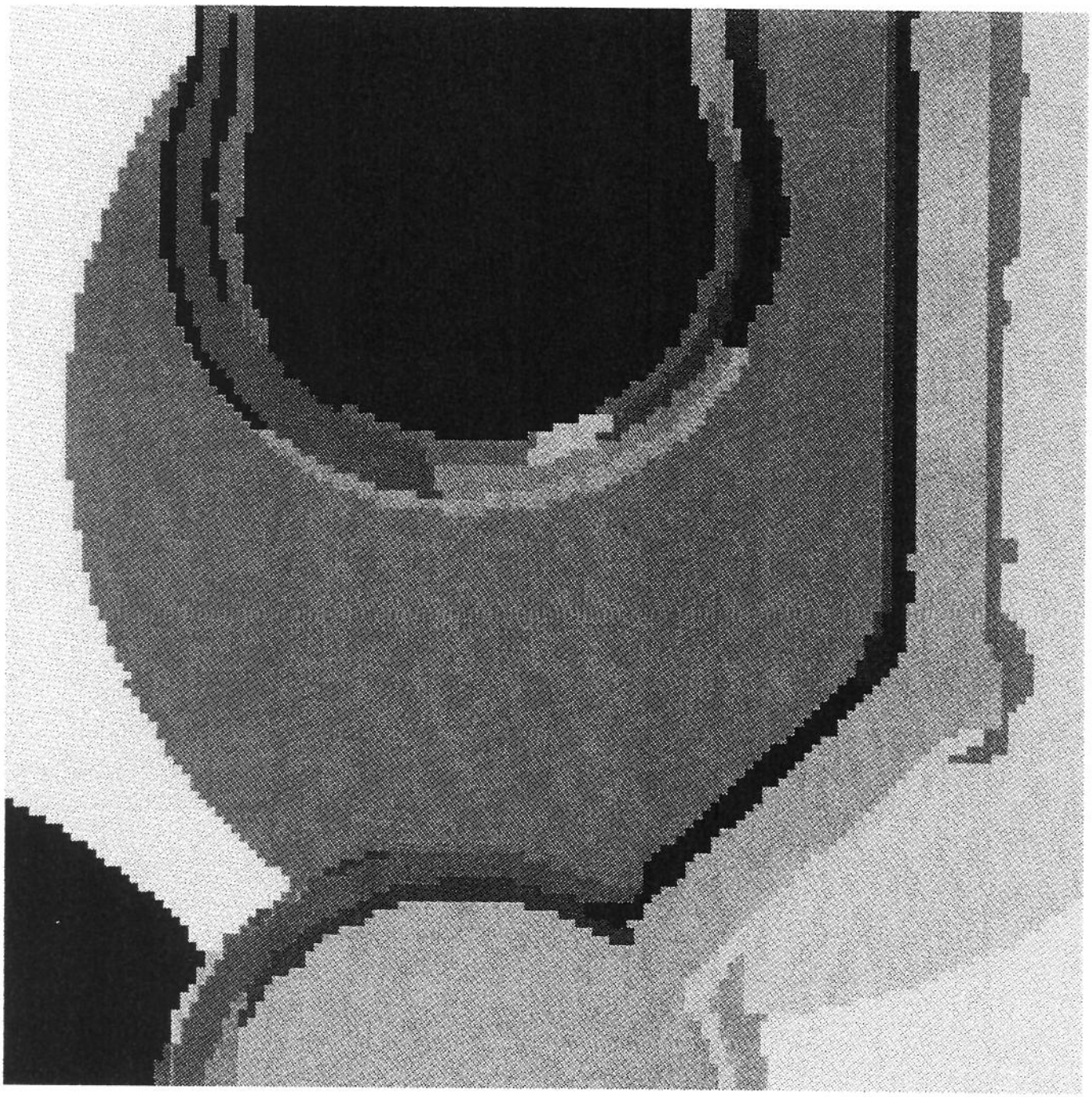

Figure 3: Segmentation result for the electronic circuit image. In this run, the maximum degree of polynomial was one. That is, a piece-wise linear model was used for each region. Notice that some of the regions in the piece-wise constant result have been merged in this result.

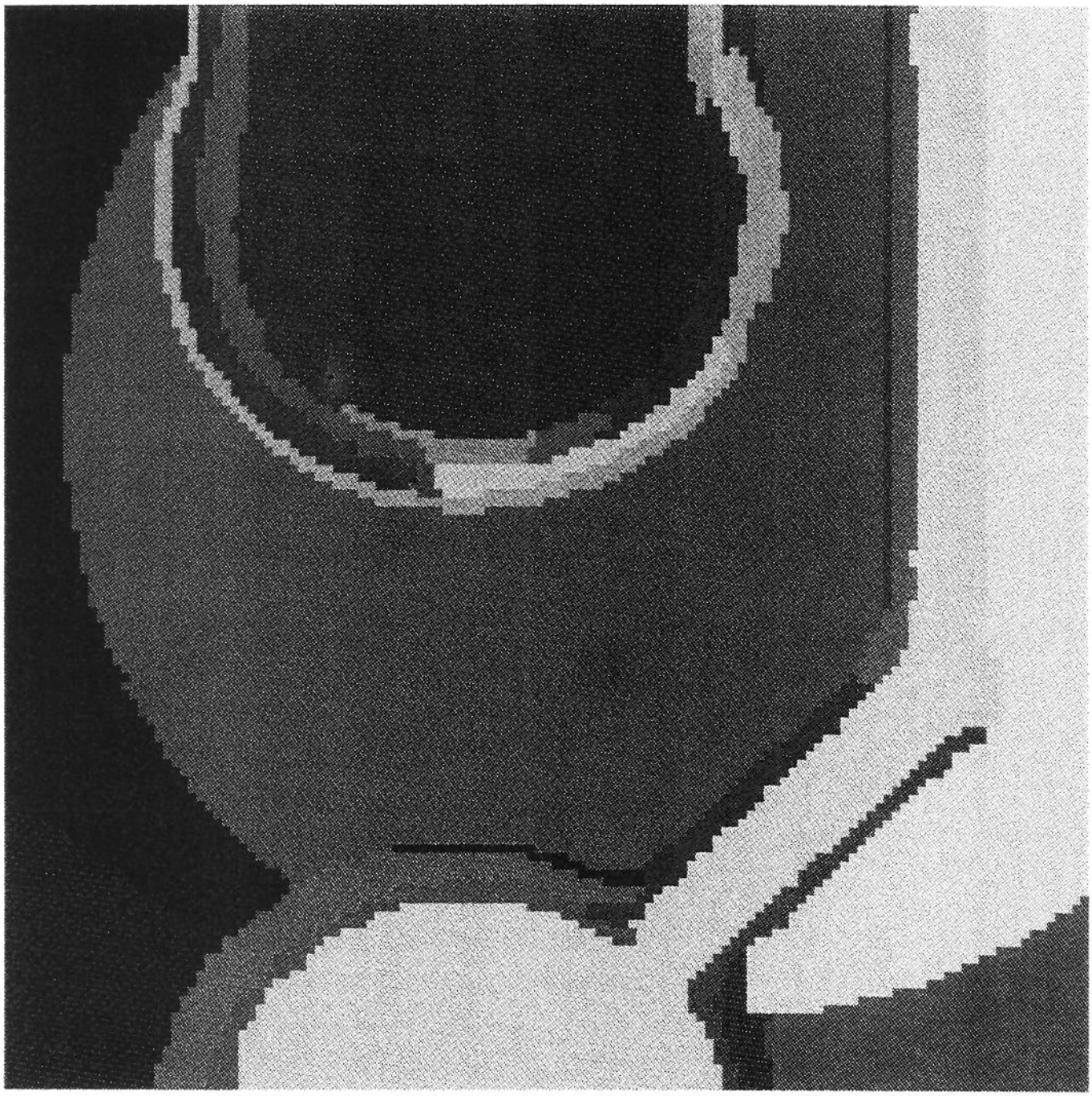

Figure 4: Segmentation result for the electronic circuit image. In this run, the maximum degree of polynomial was two. That is, a piece-wise quadratic model was used for each region. Some of the regions in the piece-wise linear result have been merged in this result.

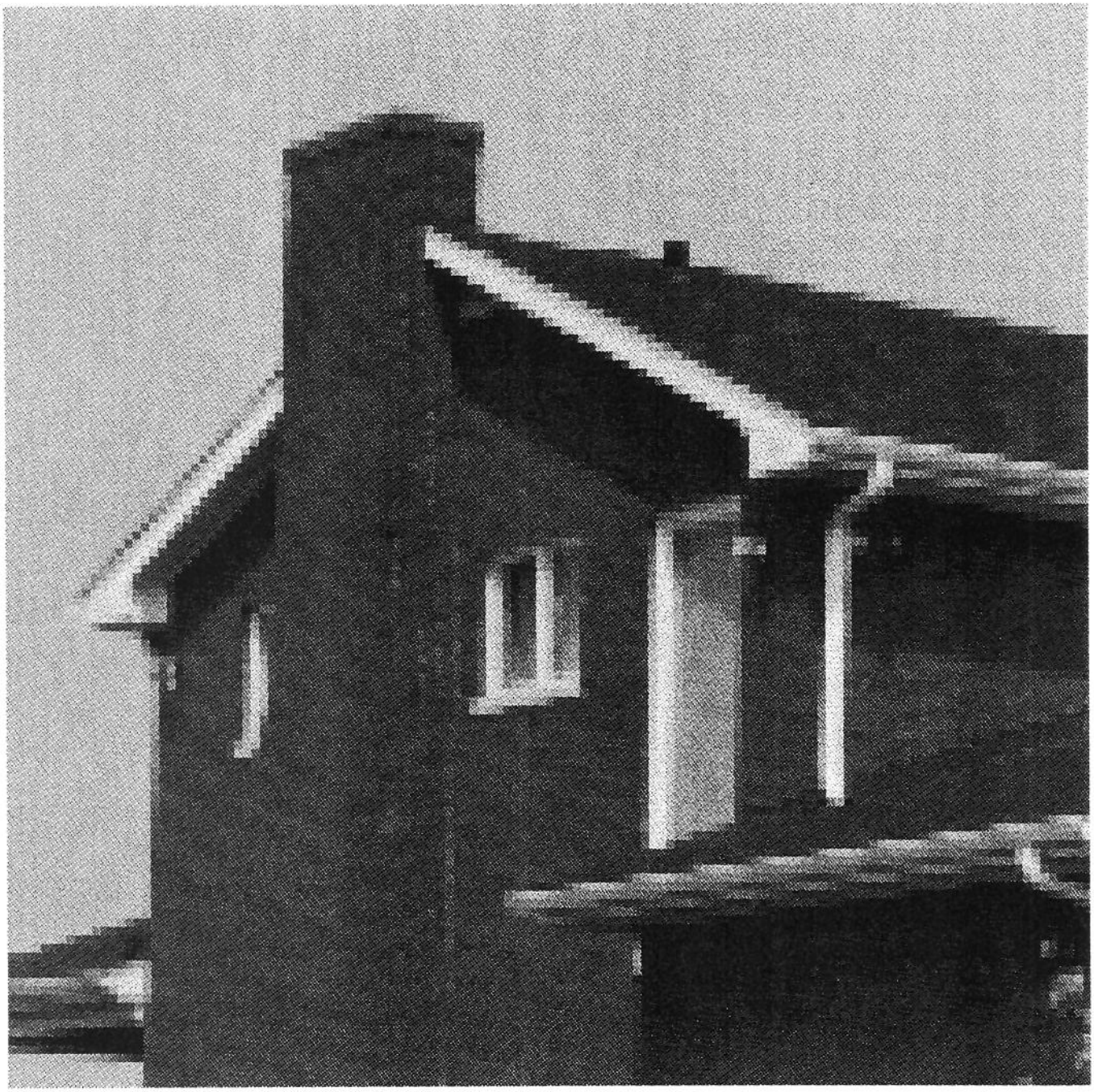

Figure 5: A real, 128 $\times$ 128, grayscale image of a house.

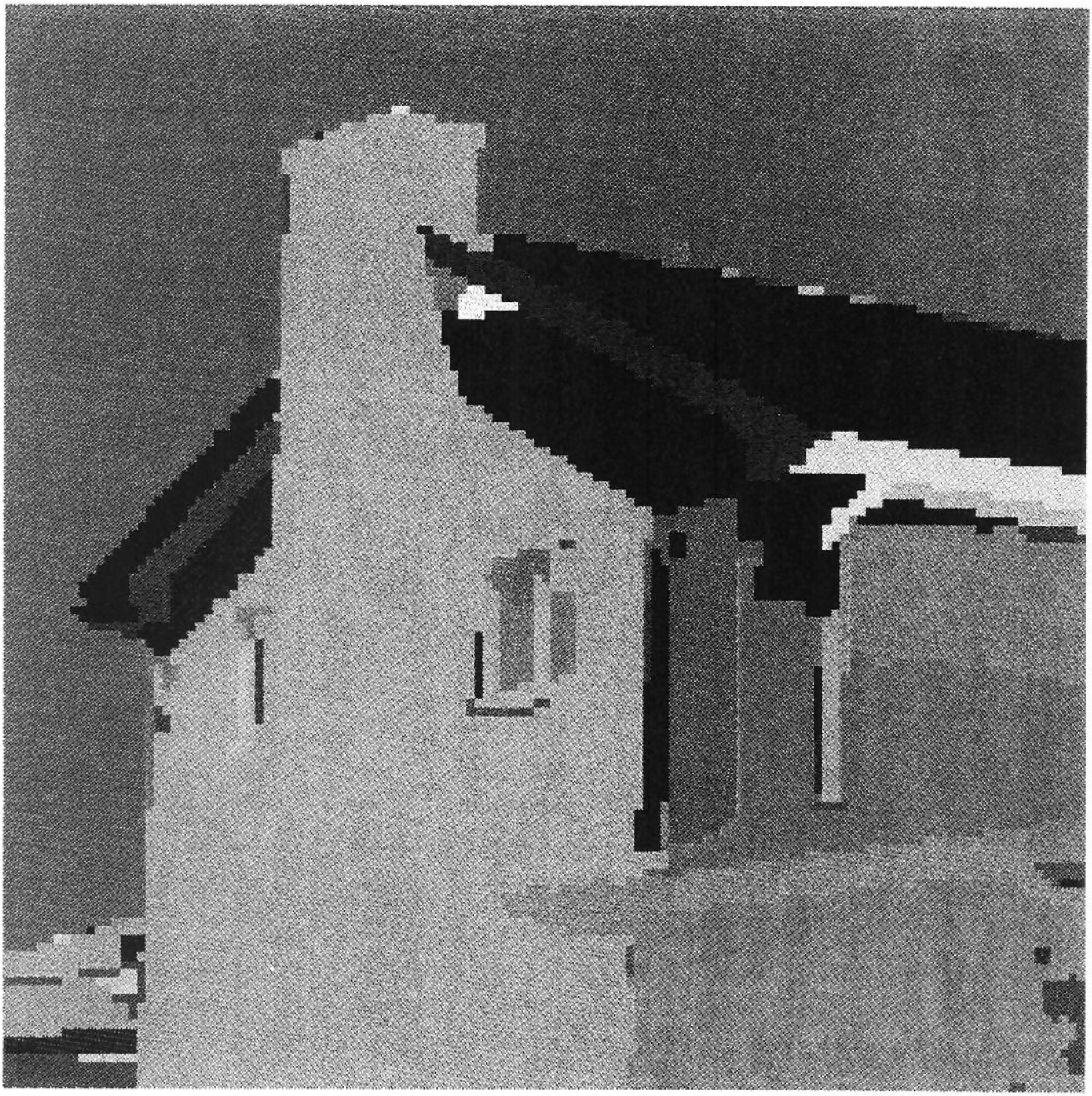

Figure 6: Segmentation result for the house image. In this run, the maximum degree of polynomial was zero. That is, a piece-wise constant model was used for each region.

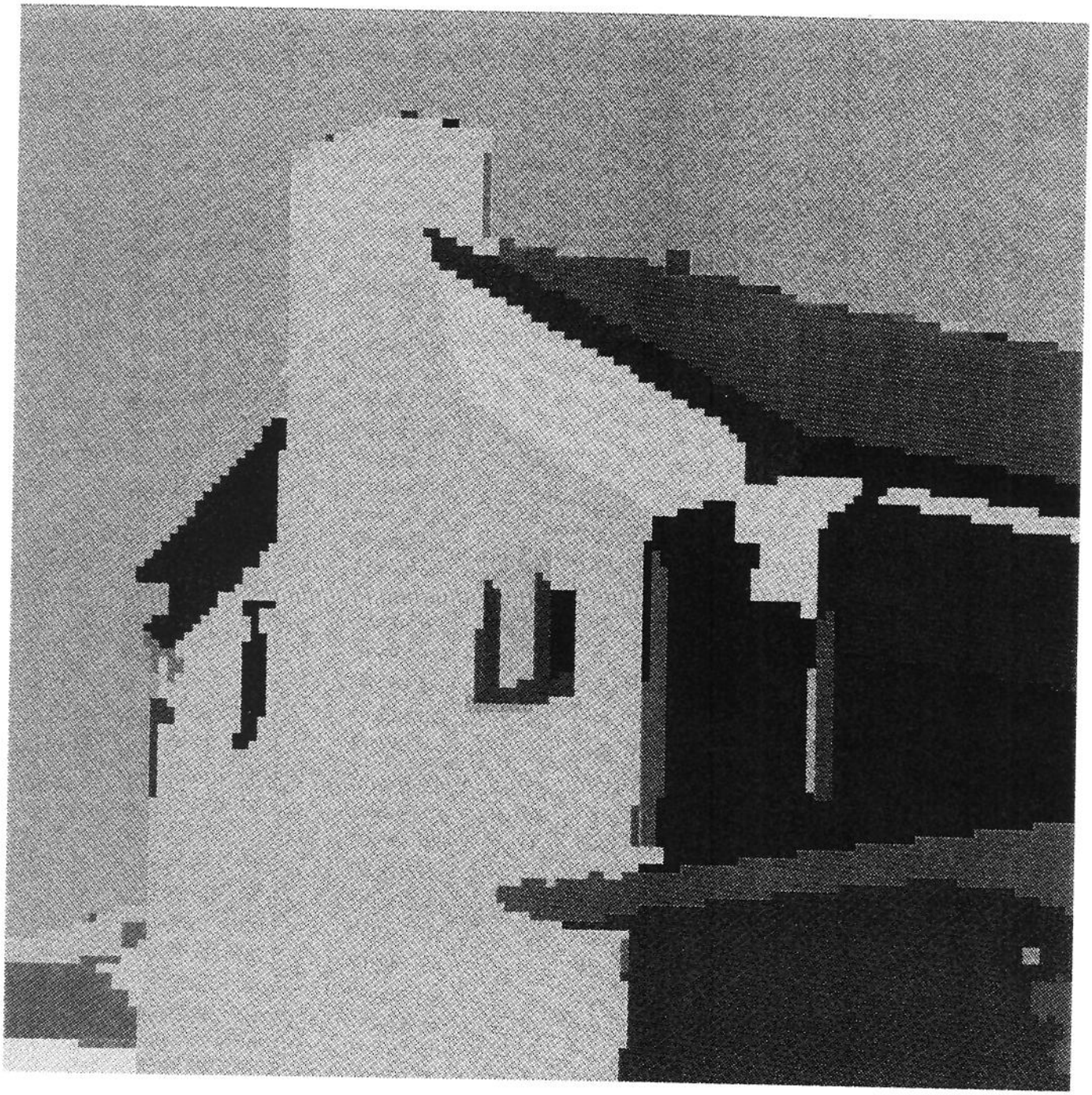

Figure 7: Segmentation result for the house image. In this run, the maximum degree of polynomial was two. Notice that some of the regions in piece-wise constant result been merged in this result.

- Recent results that obtain better (*sharper*) MDL formulas for small data sets have appeared in the in the literature[Noh93, Ris93]. Applying these for the encoding cost of the parameters, β, may produce better results. This is especially true of the results in [Noh93], which apply specifically to the problem of polynomial regression.

- Explore the possibility of using our algorithm for lossless image compression.

Acknowledgement: The authors would like to thank Jacob Shienvald, Nimrod Meggidio, Jorma Rissanen, Myron Flickner and Harpreet Sawhney for illuminating discussions.

References.

[Abr63] N. Abramson. *Information Theory and Coding.* McGraw-Hill, 1963.

[Aka74] H. Akaike. A new look at statistical model identification. *IEEE Trans.*, AC-19:716–723, 1974.

[And84] T. W. Anderson. *An Introduction to Multivariate Statistical Analysis.* Wiley, New York, second edition, 1984.

[Ber85] James O. Berger. *Statistical Decision Theory and Bayesian Analysis.* Springer-Verlag, New York, second edition, 1985.

[BG89] J. M. Beaulieu and M. Goldberg. Hierarchy in picture segmentation: A stepwise optimization approach. *IEEE PAMI*, 11(2):150–163, February 1989.

[BJ88] P. J. Besl and R. C. Jain. Segmentation through variable-order surface fitting. *IEEE Trans PAMI*, 10(2):167–192, March 1988.

[BZ87] A. Blake and A. Zisserman. *Visual Reconstruction.* MIT Press, Cambridge, MA, 1987.

[CT91] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory.* Wiley, New York, 1991.

[DP91] Trevor Darrell and Alex Pentland. Recovery of minimal descriptions using parallel robust estimation. Technical Report 163, 1991.

[DS90] Byron Dom and David Steele. 2^n−tree classifiers and realtime image segmentation. In *MVA '90: IAPR Workshop on Machine Vision Applications*, Tokyo, Japan, 1990. IAPR. (for expanded version see IBM Research Report 7558 (70424) 7/2/90).

[DSP90] Trevor Darrel, Stan Sclaroff, and Alex Pentland. Segmentation by minimal description. In *ICCV 90, Osaka, Japan*, pages 112–116, 1990.

[FH88] P. Fua and A. J. Hanson. Extracting generic shapes using model driven optimization. In *Proceedings of the Image Understanding Workshop*, pages 994–1004, Boston, 1988.

[GR80] I. S. Gradshteyn and I. M. Ryzhik. *Table of Integrals, Series, and Products.* Academic Press, New York, corrected and enlarged edition, 1980.

[Gv83] G.H. Golub and C. F. van Loan. *Matrix Computations.* The Johns Hopkins University Press, Baltimore, MD, 1983.

[HS85] R. M. Haralick and L. Shapiro. Image segmentation techniques. *Computer Vision, Graphics, and Image Processing*, 29:100–132, 1985.

[HS92] R. M. Haralick and L. G. Shapiro. *Computer and Robot Vision*, volume 1. Addison-Wesley, 1992.

[Jay83] Edwin T. Jaynes. Prior probabilities. In R. D. Rosenkrantz, editor, *E. T. Jaynes: Papers on Probability, Statistics and Statistical Physics*, chapter 7. D. Reidel, Boston, 1983. (original paper published in 1968).

[KDNS94] T. Kanungo, B. Dom, W. Niblack, and D. Steele. A fast algorithm for MDL-based multi-band image segmentation. In *International Conference on Computer Vision and Pattern Recognition.* IEEE Computer Society, 1994. to appear.

[Kee90] K. Keeler. Minimal length encoding of planar subdivision topologies with application to image segmentation. In *AAAI 1990 Spring Symposium of the Theory and Application of Minimal Length Encoding*, 1990.

[KMWP90] Daniel Keren, Ruth Marcus, Michael Werman, and Shmuel Peleg. Segmentation by minimum length encoding. In *ICPR 90*, pages 681–683, 1990.

[LCJ91] S. Liou, A. H. Chin, and R. Jain. A parallel technique for signal-level perceptual organization. *IEEE Trans PAMI*, 13(4):317–325, 1991.

[Lec89a] Y. G. Leclerc. Constructing simple stable descriptions for image partitioning. *International Journal of Computer Vision*, 3:73–102, 1989.

[Lec89b] Y. G. Leclerc. *The Local Structure of Image Intensity Discontinuities.* PhD thesis, McGill University, Montréal, Québec, Canada, May 1989.

[Lec90] Y. G. Leclerc. Region grouping using the minimum-description-length principle. In *DARPA Image Understanding Workshop*, 1990.

[Liu77] C. L. Liu. *Elements of Discrete Mathematics.* McGraw Hill, New York, 1977.

[LJ91] S. Liou and R. Jain. An approach to three-dimensional image segmentation. *CVGIP: Image Understanding*, 53(3):237–252, 1991.

[Noh93] Ragnar Nohre. *Topics in Descriptive Complexity.* PhD thesis, Technical University of Linkoping, 1993. See Chapt. 2 Coding Small Data Sets.

[Pen89] Alex Pentland. Part segmentation for objects recognition. *Neural Computation*, 1:82–91, 1989.

[PTaBPF92] W. H. Press, S. A. Teukolsky, and W. T. Vetterling and B. P. Flannery. *Numerical Recipes in C (2 ed.).* Cambridge University Press, 1992.

[Ris78] J. Rissanen. Modelling by shortest data description. *Automatica*, 14:465–471, 1978.

[Ris83] J. Rissanen. A universal prior for integers and estimation by minimum description length. *The Annals of Statistics*, 2(11):211–222, 1983.

[Ris89] J. Rissanen. *Stochastic Complexity in Statistical Inquiry*, volume 15. World Scientific Series in Computer Science, 1989.

[Ris93] Jorma Rissanen. Fisher information and stochastic complexity. Research Report RJ 9547, IBM Research Division, 1993.

[SDNS92] J. Sheinvald, B. Dom, W. Niblack, and D. Steele. Unsupervised image segmentation using the minimum description length principle. In *Proceedings of ICPR 92*, August 1992. For an expanded version see: *IBM Research Report RJ 8474 (76695)*, (11/1/91).

[Sed90] R. Sedgewick. *Algorithms in C.* Addison Wesley, 1990.

[Sha48] C. E. Shannon. A mathematical theory of communication. *Bell Syst Tech J.*, (3):379–423, 1948.

[Str80] G. Strang. *Linear Algebra and its Applications.* Academic Press, New York, NY, 1980.

[WK90] R. S. Wallace and T. Kanade. Finding natural clusters having minimum description length. In *AAAI 1990 Spring Symposium on the Theory and Application of Minimum Length Methods*, Stanford University, Stanford, CA, 1990.

[ZM90] J. Zhang and J. W. Modestino. A model fitting approach to cluster validation with application to stochastic model-based image segmentation. *IEEE PAMI*, 12(10):1009–1017, October 1990.

A. MDL Principle for Estimation

Here we present a brief tutorial discussion of the Minimum Description Length principle (MDL)[Ris78] for estimation or model selection. It has been described in depth in [Ris89] and an interesting historical account of its development is presented in [Lec89b]. The MDL principle addresses the following general problem: Given a set of observations $\mathbf{Y} = \{y_1, \ldots, y_n\}$ (e.g. a set of pixel measurement vectors corresponding to a greyscale or multiband image) and a set of competing candidate models $\{M \in \mathcal{M}\}$, select the best model, where, in principle, "best" refers to the purpose for which the model is to be used; for example (in our case) for unsupervised image segmentation. The MDL approach defines as *best* the model that yields the shortest code length for the given observations, when used in an ideal coding-theoretic scheme. It is based on information-theoretic arguments, and can (as mentioned in the body of this paper) be viewed as a formalization of *Ockham's razor:* the simplest model explaining a set of observations is the best[15].

Much of the work on this topic by Rissanen[Ris89] and others might be described as addressing the issue of how one formalizes this minimum description length principle mathematically. This formalization is achieved by building on the work of

[15] Attributed to William of Ockham (1285-1349). The exact quote usually given is: "entities are not to be multiplied beyond necessity". This is also referred to as the *law of economy* or the *law of parsimony.*

Shannon [Sha48] and others (information theory [Abr63, CT91]), associating codelengths with probabilities. The MDL criterion we use, which is based on a two-part coding scheme, represents one such formalization. We derive this criterion as follows. One envisions a scheme that uses a class of parametric probability models, $\{p(\mathbf{Y}|M)\}$ ("indexed" by M) to encode the data, $\mathbf{Y}$. The data is encoded by first encoding the model and then encoding the data using ("conditioned on") the model. Thus the total codelength is given by: $L(M) + L(\mathbf{Y}|M)$. The codelength we use for the data (2nd term) is given by[16]: $L(\mathbf{Y}|M) = -\log p(\mathbf{Y}|M)$. [17] There is a certain redundancy hidden in this coding formulation, however. This can be seen as follows. The model, M is selected to minimize this codelength. However, only a subset of the possible $\mathbf{Y}$ values could have resulted in the selection of a particular M, whereas this codelength implies a code that would allow any $\mathbf{Y}$ to be encoded using M, even one that couldn't possibly have led to its selection.

To correct this redundancy we can, in principle , given M, construct a normalized density over only these allowable $\mathbf{Y}$ values by simply dividing $p(\mathbf{Y}|M)$ by $\sum_{\mathbf{Y}\in R(M)} p(\mathbf{Y}|M)$, where $R(M)$ is the set of allowable $\mathbf{Y}$ values. Then a code whose lengths are given approximately by logarithms of this normalized density can be constructed. Because of the problem in computing this sum, however, we ignore it and (as has been the usual practice) thus use a somewhat less efficient encoding scheme, which (one would assume) will result in slightly worse estimates. Recent results in this area[Noh93, Ris93] would appear to improve on this criterion by addressing this redundancy and other issues. In future work we will utilize these new results. Here we will only describe and use the older results, however.

The issue of how to encode the model is not quite as straightforward, as that of encoding the data using the model, however. If $\mathcal{M}$ is a discrete set (either finite or countably infinite)[18] and we have an associated *prior*[19] $p(M)$, we may simply use it to

[16] A minor point is being swept under the rug here. Throughout our development we will treat $\mathbf{Y}$ as real-valued and $p(\mathbf{Y}|M)$ as a continuous probability *density*. The coding paradigm applies only to discrete "symbols" however. Implicit in our development is that the scalar values composing $\mathbf{Y}$ have been truncated to the appropriate precision and can, therefore, be represented by integers. It would therefore be more correct to write $L(\mathbf{Y}|M) = -\log[p(\mathbf{Y}|M) \cdot \delta\mathbf{Y}]$, where (because y values are integers) $\delta\mathbf{Y} = 1$.

[17] The notation used here might seem a little confusing when first encountered. The variable M may be thought of as a specification of the model and since we follow the Bayesian notion of considering this to be a random variable, we use the *conditional* notation, $(\ldots|\ldots)$. In this context, $p(\mathbf{Y}|M)$, denotes the value of the probability density one obtains for $\mathbf{Y}$ using the model specified by M.

[18] The decision trees of [DS90] for example.

[19] This term is used in the paradigm of Bayesian estimation[Ber85] where the model M (or its

accomplish this encoding (i.e. $L(M) = -\log p(M)$). If we have no such prior, then one must be constructed. Techniques have been developed for constructing priors using limited prior knowledge. For example see: [Ber85] and [Jay83]. In cases where there is essentially no prior knowledge, Rissanen's *Universal Prior for Integers* may be used[20]. It should be pointed out that in this discrete-model case constructing a prior and constructing an encoding scheme are equivalent; the length of the latter being essentially the logarithm of the former.

In the more common case the model specification, M, is divided into a *structure parameter* (usually integer-valued), κ (which indexes the discrete part of the model), and a vector of real-valued parameters[21] $\beta^{(\kappa)}$, where the dimensionality of $\beta^{(\kappa)}$ is $K^{(\kappa)}$. As a simple example consider the problem of polynomial (in one variable) curve fitting. In this case κ is the polynomial order, $K^{(\kappa)} = \kappa + 1$, and $\beta^{(\kappa)}$ is the set of $\kappa+1$ coefficients. In our image-segmentation application κ specifies the partitioning (segmentation) of the image into regions and the orders of the 2D polynomial surfaces for all the regions, while the components of $\beta^{(\kappa)}$ are the polynomial coefficients and the variances and covariances (components of the covariance matrix).

When the model structure is fixed, the two most common criteria for estimating β are *maximum likelihood* (ML) (choosing β to maximize $p(\mathbf{Y}|\beta)$) and the Bayesian *maximum a posteriori* (MAP) estimation (choosing β to maximize the *posterior* density: $p(\beta|\mathbf{Y}) = p(\mathbf{Y}, \beta)/p(\mathbf{Y})$). From the coding perspective assumed by MDL, however, these estimates beg the question of what precision is to be used in encoding β. Since β is real-valued, infinite precision is required (in general) to represent it with complete accuracy. This problem is, in a sense, an opportunity for the MDL principle to exert its influence. Rissanen has developed a technique for determining

---

specification) is considered to be a random variable with its own probability distribution, $P(M)$, usually referred to as the *prior* because it presumably embodies prior knowledge about the likelihood of various values of M. Occasionally, there is a known process characterizable by a known distribution, $P(M)$. In the more usual situation, $P(M)$, is not known precisely. In such cases it is chosen to be consistent with whatever is known (both qualitatively and quantitatively) and, within those constraints, in the most computationally convenient form.

[20]This provides a prior (or, equivalently, a coding scheme) for the positive integers. If the models of the discrete set $\mathcal{M}$ have no natural ordering, then their association with the positive integers is arbitrary and this creates a minor problem because Rissanen's prior slightly favors small integers. If the set $\mathcal{M}$ is countably infinite (the usual case for using Rissanen's prior), then it almost certainly has a natural ordering and (if this isn't already the case) its indexing should be transformed such that the complexity of the models increases with the index.

[21]In this tutorial section we use the superscript (κ) to remind the reader that the dimensionality of β depends on the model structure κ. In the main body of the paper, we drop this superscript and the symbol, κ, for ease of notation.

the optimal precision for encoding real-valued parameters[Ris83]. This technique is based on a universal scheme for encoding discretized real-valued vectors. The codelengths associated with this scheme cannot be considered negative logarithms of a prior (in the Bayesian sense), because the codes and associated codelengths depend on the data, $\mathbf{Y}$. For the case of ML estimation (i.e. no prior $p(\beta)$ is available), when this is taken to a suitable asymptotic ($n \rightarrow \infty$)[22] form it may be written as:[23]

$$L(\hat{\beta}^{(\kappa)}|\kappa, \mathbf{Y}) = \tfrac{1}{2} K^{(\kappa)} \log n \tag{A.1}$$

where $\hat{\beta}^{(\kappa)}$ is the maximum likelihood estimator of $\beta^{(\kappa)}$ (i.e. the minimizing value for $-\log p(\mathbf{Y}|\beta^{(\kappa)})$) and $K^{(\kappa)}$ is the number of "free" components in $\beta^{(\kappa)}$. Note that the only remaining dependence on the data in equation (A.1) is on the number of data points n.

For the case of MAP estimation we have:

$$L(\hat{\beta}^{(\kappa)}|\kappa) = -\log p(\hat{\beta}^{(\kappa)}) \quad + \quad \tfrac{1}{2} K^{(\kappa)} \log n \tag{A.2}$$

In this case $\hat{\beta}^{(\kappa)}$ is the MAP estimate, which will, in general, be different from the ML estimate. The $\frac{1}{2} K^{(\kappa)} \log n$ term is usually called the "penalty term", since it penalizes models with many parameters. Some workers use the term "MDL criterion" to refer to the ML form of the minimum description length, using this asymptotic form for the description length of β. This may be written as:

$$MDL(\kappa) = -\log p(\mathbf{Y}|\hat{\beta}^{(\kappa)}) + \tfrac{1}{2} K^{(\kappa)} \log n + L(\kappa) \tag{A.3}$$

where $L(\kappa)$ is the encoding length for κ based on whatever encoding scheme is used for the particular problem. In some cases (e.g. polynomial curve fitting) $L(\kappa)$ is dropped because it is either the same for all κ or insignificant. In our image-segmentation application we ignore the polynomial orders and κ reduces to $\mathbf{\Omega}$, the segmentation of the image.

[22] The recent results mentioned above[Noh93, Ris93] also eliminate the parameter-precision issue and offer an expression valid for small n.

[23] There is a subtle flaw in this notation, but it is fine for our purposes. Using this term, the total code length is given by: $L(\mathbf{Y}|\hat{\beta}^{(\kappa)}) + L(\hat{\beta}^{(\kappa)}|\kappa, \mathbf{Y})$. This seems to imply that the value of $\hat{\beta}^{(\kappa)}$ is used to encode the data and that $\hat{\beta}^{(\kappa)}$ is then encoded with $L(\hat{\beta}^{(\kappa)}|\kappa, \mathbf{Y})$ bits. In reality, of course, the optimal-precision approximation to $\hat{\beta}^{(\kappa)}$ is used to encode the data, $\mathbf{Y}$. For an in-depth discussion see Rissanen's original paper [Ris83] or the discussion on pp. 54-58 of his book [Ris89].

B. A Discussion of the Encoding Scheme for Region Boundaries

One might be concerned about an apparent inefficiency in chain-code-based boundary (image-partition) encoding scheme we use. There is a subtle point here. This encoding scheme allows the description of impossible partitionings (segmentations) (e.g. boundary segments that intersect themselves, go outside the image, etc.) and is thus inefficient from the viewpoint of coding. If this coding scheme were used directly as part of a practical scheme for image compression, this inefficiency would be real. In doing model selection (estimation), however, we need not be concerned with this kind of inefficiency for the following reason. Let $L(\mathbf{\Omega})$ represent the length function for our encoding scheme. L is defined for all valid segmentations. Using the Kraft inequality[Abr63], we can construct a probability function $P(\mathbf{\Omega})$ as follows:

$$p(\mathbf{\Omega}) \;=\; 2^{-L(\mathbf{\Omega})} / \sum_{\mathbf{\Omega}} 2^{-L(\mathbf{\Omega})}, \tag{B.1}$$

where the sum is over all possible segmentations. Using p, we may now, in principle, construct a new coding scheme with code lengths given approximately by: $-\log p(\mathbf{\Omega})$, which means that they will differ only by an additive constant (the logarithm of denominator in the RHS of equation (B.1)) from $L(\mathbf{\Omega})$. Because this constant is the same for all models it will have no effect on which model (i.e. which segmentation) we select and no harm comes, therefore, from omitting it and using our original $L(\mathbf{\Omega})$.

This reasoning may be misleading, however. Suppose we were to construct codes for $\mathbf{\Omega}$ that linearly increase with boundary length, but are longer than the ones we've described here. That is, the lengths of these new codes are equal to a constant, ($\alpha > 1$), times the ones described here. In that case the derivative of our code length with respect to the boundary length of $\mathbf{\Omega}$ is $\alpha \log_2 3$. Thus, the larger we make α the greater the degree to which our objective function will favor segmentations with a shorter total boundary length. If we think of this $p(\mathbf{\Omega})$ as a *prior* probability in the Bayesian sense, increasing α is equivalent to concentrating more of the probability in the region of shorter boundary lengths. The probability associated with the zero-boundary-length segmentation, $\mathbf{\Omega}_0$, (i.e. a single region that is the entire image) is given by:

$$p(\mathbf{\Omega}_0) \;=\; \left[\sum_{\mathbf{\Omega}} 2^{-L(\mathbf{\Omega})}\right]^{-1}, \tag{B.2}$$

An alternate coding scheme that is possible in principle is to specify the total boundary length corresponding to $\mathbf{\Omega}$, $l(\mathbf{\Omega})$, and then to specify which of the possible

partitionings with that total boundary length is used. This can be done by ordering all the partitionings for a given l in any arbitrary way (the encoder and decoder just need to use the same ordering) and then specifying the partitioning by its index (position in the list). The number of bits required to specify this index is the logarithm of the number of possible partitionings with the specified total boundary length.

This has no inefficiency of the type described above (i.e. allowing the description of impossible boundaries), but is less tractable because it cannot be written as a sum of terms, each corresponding to a single boundary segment, and therefore doesn't lend itself as well to our region-merging optimization scheme because the codelength change associated with merging two regions (and thus eliminating their common boundary segment) depends on the boundary length of the entire segmentation and not just that of shared segment as is the case for our chain-code-based scheme. This scheme also has the problem that is has a maximum with respect to codelength that doesn't correspond to the maximum boundary length. In fact, the codelength for a complete "checker board" (where every pixel is a separate region) is the same as that for Ω_0 because in each case the number of possible partitionings with the corresponding total boundary length is one. This maximum is a problem for us, because we have chosen total boundary length as our measure of complexity.

C. A proof for equivalence of two solutions

In this appendix we show that regression solutions obtained by matrix inverse and by solving a linear system of equations are one and the same.

First we need the following lemma (from Strang [Str80]).

Lemma C.1. Let $\mathbf{\Phi}$ be a $N \times M$ full rank matrix with $N > M$. That is, $rank(\mathbf{\Phi}) = M$. Then $rank(\mathbf{\Phi}^t\mathbf{\Phi}) = M$, and thus $\mathbf{\Phi}^t\mathbf{\Phi}$ is invertible and has the null space $\{0\}$.

Proof:
We prove the lemma by (i) showing that the null space of $\mathbf{\Phi}$ and the null space of $\mathbf{\Phi}^t\mathbf{\Phi}$ are of same dimensions, and (ii) using the fact that the number of colums of $\mathbf{\Phi}$ and $\mathbf{\Phi}^t\mathbf{\Phi}$ are the same.

Let $\mathbf{x}$ be in the null space of $\mathbf{\Phi}$. Therefore $\mathbf{\Phi x} = 0$. Multiplying by $\mathbf{\Phi}^t$ we have $\mathbf{\Phi}^t\mathbf{\Phi x} = 0$. That is, if $\mathbf{x}$ is in the null space of $\mathbf{\Phi}$, then it is in the null space of $\mathbf{\Phi}^t\mathbf{\Phi}$. Now the converse. Let $\mathbf{\Phi}^t\mathbf{\Phi x} = 0$. Then, if we take the inner product with $\mathbf{x}$ we

have, $\mathbf{x}^t\mathbf{\Phi}^t\mathbf{\Phi}\mathbf{x} = 0$. That is, $||\mathbf{\Phi}\mathbf{x}||^2 = 0$, or, $\mathbf{\Phi}\mathbf{x} = 0$. Thus $\mathbf{\Phi}$ and $\mathbf{\Phi}^t\mathbf{\Phi}$ have the same null space.

But $M = rank(\mathbf{\Phi}) = M - dim(\text{null space of } \mathbf{\Phi}) = M - dim(\text{null space of } \mathbf{\Phi}^t\mathbf{\Phi}) = rank(\mathbf{\Phi}^t\mathbf{\Phi})$. Thus, the rank of $\mathbf{\Phi}^t\mathbf{\Phi}$ is M and hence it is invertible and has the null space $\{0\}$. ∎

Proposition C.1. Let $\mathbf{\Phi}$ be a $N \times M$ full rank matrix with $N > M$. That is, $rank(\mathbf{\Phi}) = M$. Then if $\hat{\boldsymbol{\theta}}$ minimizes

$$||\mathbf{y} - \mathbf{\Phi}\boldsymbol{\theta}||^2,$$

then $\hat{\boldsymbol{\theta}}$ also minimizes

$$\left|\left|\mathbf{\Phi}^t\mathbf{y} - \mathbf{\Phi}^t\mathbf{\Phi}\boldsymbol{\theta}\right|\right|^2.$$

Proof:
Now, $\hat{\boldsymbol{\theta}}$ minimizes the fitting error if and only if the error vector is orthogonal to the space spanned by the the basis vectors. Thus, $\hat{\boldsymbol{\theta}}$ minimizes $||\mathbf{\Phi}^t\mathbf{y} - \mathbf{\Phi}^t\mathbf{\Phi}\boldsymbol{\theta}||^2$ if and only if $(\mathbf{\Phi}^t\mathbf{\Phi})^t(\mathbf{\Phi}^t\mathbf{y} - \mathbf{\Phi}^t\mathbf{\Phi}\hat{\boldsymbol{\theta}}) = 0$. But since $\mathbf{\Phi}$ is full rank, the null space of $\mathbf{\Phi}^t\mathbf{\Phi}$ is $\{0\}$ (from Lemma A.1). This implies that $\hat{\boldsymbol{\theta}}$ minimizes $||\mathbf{\Phi}^t\mathbf{y} - \mathbf{\Phi}^t\mathbf{\Phi}\boldsymbol{\theta}||^2$ if and only if $\mathbf{\Phi}^t\mathbf{y} - \mathbf{\Phi}^t\mathbf{\Phi}\hat{\boldsymbol{\theta}} = 0$. Factoring out $\mathbf{\Phi}^t$ we have $\mathbf{\Phi}^t(\mathbf{y} - \mathbf{\Phi}\hat{\boldsymbol{\theta}}) = 0$. But this is true if and only if $\hat{\boldsymbol{\theta}}$ minimizes $||\mathbf{y} - \mathbf{\Phi}\boldsymbol{\theta}||^2$. Thus we have proved our claim. ∎

D. Computation of $-\log p(\mathbf{Y}_j|\hat{\beta}_j)$

In this appendix we will prove some of the results used in the section on residual encoding. First, we will need the following proposition which represents the exponent of a multivariate Gaussian maximum likelihood formulation in terms of sample covariance matrices.

Proposition D.1. Let $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n$ and $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_n$ (in our application/notation $\boldsymbol{\mu}_i \equiv \boldsymbol{\mu}(\mathbf{x}_i)$) be n d-dimensional vectors and let $\mathbf{\Sigma}$ be a positive definite matrix. Then,

$$\sum_{i=1}^{n}(\mathbf{y}_i - \boldsymbol{\mu}_i)^t\mathbf{\Sigma}^{-1}(\mathbf{y}_i - \boldsymbol{\mu}_i) = n \cdot \text{trace}\left\{\mathbf{\Sigma}^{-1}\mathbf{S}\right\}$$

where

$$\mathbf{S} = \sum_{i=1}^{n} (\mathbf{y}_i - \boldsymbol{\mu}_i)(\mathbf{y}_i - \boldsymbol{\mu}_i)^t \ .$$

Proof:
Let y_{ij} be the j^{th} component of the vector $\mathbf{y}_i$. Similarly, let μ_{ij} represent the j^{th} component of the vector $\boldsymbol{\mu}_i$. Now we will try to simplify the left had side and the right hand side of the proposition and show that they are the same.

$$\begin{aligned} L.H.S. &= \sum_{i=1}^{n} (\mathbf{y}_i - \boldsymbol{\mu}_i)^t \boldsymbol{\Sigma}^{-1} (\mathbf{y}_i - \boldsymbol{\mu}_i) \\ &= \sum_{i=1}^{n} \sum_{j=1}^{d} \sum_{k=1}^{d} (y_{ij} - \mu_j) \sigma_{jk}^{-1} (y_{ik} - \mu_k) \ . \end{aligned} \tag{D.1}$$

Now, s_{ij}, the $(i,j)^{th}$ component of the matrix $\mathbf{S}$ can be computed as

$$s_{jk} = \frac{1}{n} \sum_{i=1}^{n} (y_{ij} - \mu_{ij})(y_{ik} - \mu_{ik}) \ .$$

We will use the above equation to simplify the right hand side of the proposition.

$$\begin{aligned} R.H.S. &= n \cdot \text{trace} \left\{ \boldsymbol{\Sigma}^{-1} \mathbf{S} \right\} \\ &= n \sum_{j=1}^{d} \sum_{k=1}^{d} \sigma_{jk}^{-1} s_{jk} \\ &= n \sum_{j=1}^{d} \sum_{k=1}^{d} \sigma_{jk}^{-1} \sum_{i=1}^{n} (y_{ij} - \mu_{ij})(\mathbf{y}_{ik} - \mu_{ik}) \\ &= \sum_{i=1}^{n} \sum_{j=1}^{d} \sum_{k=1}^{d} (y_{ij} - \mu_{ij}) \sigma_{jk}^{-1} (y_{ik} - \mu_{ik}) \\ &= L.H.S. \ . \end{aligned}$$

Thus proved. ∎

Proposition D.2. Let $\mathbf{Y} = [\mathbf{y}_1 \mathbf{y}_2 \cdots \mathbf{y}_n]^t$ where $\mathbf{y}_i \in R^d$, and let $\mathbf{y}_i$ be samples from the Gaussian distribution[24] $N(\mathbf{\Phi\Theta}, \mathbf{\Sigma})$, where $\mathbf{\Phi}$ is a constant matrix. Then

$$-\log p(\mathbf{Y}|\mathbf{\Phi}\hat{\mathbf{\Theta}}, \hat{\mathbf{\Sigma}}) = \frac{1}{2}dn(1 + \log 2\pi) + \frac{1}{2}n\left|\hat{\mathbf{\Sigma}}\right| \ .$$

Proof:
Since $\mathbf{y}$ is Gaussian distributed, we have

$$-\log p(\mathbf{Y}|\mathbf{\Phi\Theta}, \mathbf{\Sigma}) = \frac{n}{2}(d\log 2\pi + |\mathbf{\Sigma}|) + \frac{1}{2}\sum_{i=1}^{n}(\mathbf{y}_i - \mathbf{\Phi\Theta})^t\mathbf{\Sigma}^{-1}(\mathbf{y}_i - \mathbf{\Phi\Theta}) \ .$$

From proposition D.1 we have

$$-\log p(\mathbf{Y}|\mathbf{\Phi\Theta}, \mathbf{\Sigma}) = \frac{n}{2}(d\log 2\pi + \log|\mathbf{\Sigma}|) + \frac{n}{2}\cdot \text{trace}\left\{\mathbf{\Sigma}^{-1}\mathbf{S}\right\} \ ,$$

where $\mathbf{S} = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{y}_i - \mathbf{\Phi\Theta})(\mathbf{y}_i - \mathbf{\Phi\Theta})^t$. But, $\hat{\mathbf{\Theta}}$ and $\hat{\mathbf{\Sigma}}$ minimize $-\log p(\mathbf{Y}|\mathbf{\Phi\Theta}, \mathbf{\Sigma})$. Substituting for $\mathbf{\Phi}\hat{\mathbf{\Theta}}$ for $\mathbf{\Phi\Theta}$ in the computation of $\mathbf{S}$ and $\hat{\mathbf{\Sigma}}$ for $\mathbf{\Sigma}$, we have,

$$\begin{aligned} -\log p(\mathbf{Y}|\mathbf{\Phi}\hat{\mathbf{\Theta}}, \hat{\mathbf{\Sigma}}) &= \frac{n}{2}(d\log 2\pi + \log\left|\hat{\mathbf{\Sigma}}\right|) + \frac{n}{2}\cdot \text{trace}\left\{\hat{\mathbf{\Sigma}}^{-1}\hat{\mathbf{\Sigma}}\right\} \\ &= \frac{n}{2}(d\log 2\pi + \log\left|\hat{\mathbf{\Sigma}}\right|) + \frac{n}{2}\cdot d \\ &= \frac{1}{2}dn(1 + \log 2\pi) + \frac{1}{2}n\log\left|\hat{\mathbf{\Sigma}}\right| \end{aligned}$$

Thus proved. ∎

E. Number of terms in a k^{th}-degree polynomial in q variables

In this appendix we compute an expression for the number of terms in a k^{th}-degree polynomial in q variables. This is necessary in order to compute the surface regression, and the encoding cost of the model. The q in our context is the dimensionality of the pixel coordinates, and not the number of bands, d of the image. For example,

[24]The appearance of $\mathbf{\Phi\Theta}$ as the mean in the Gaussian distribution, $N(\mathbf{\Phi\Theta}, \mathbf{\Sigma})$ might warrant some explanation. This corresponds to our position-dependent non-stationary mean, $\mu(\mathbf{x})$. The $(i,j)^{th}$ element of $\mathbf{\Phi\Theta}$ is the μ value for the $(i,j)^{th}$ element of $\mathbf{Y}$.

for CT volume data $q = 3$. As usual, k represents the degree of the polynomial that is fit to the data in each band.

We will use the method of generating functions for the computation (see [Liu77]). Notice that the number of terms with degree j in a k^{th}-degree polynomial ($j \leq k$) in q variables $x_1, \ldots, x_q$, is given by the coefficient of z^j in the expansion[25]

$$A(z) = (1 + z + \cdots + z^k)^q \ .$$

This is because the coefficient of z^j is the number of ways we can make up z^j as a product of q factors taken from the set $\{1, z, \ldots, z^k\}$. Now, putting $A(z)$ in a form where it is possible to isolate the z^j term, we use the partial summation formula to obtain:

$$\begin{aligned}
(1 + z + \cdots + z^k)^q &= \left(\frac{1 - z^{k+1}}{1 - z}\right)^q \\
&= (1 - z)^{-q} \cdot (1 - z^{k+1})^q \\
&= \left[\sum_{l=0}^{\infty} \binom{q + l - 1}{l} (-1)^l z^l\right] \cdot \left[\sum_{i=0}^{q} \binom{q}{i} (-1)^i (z^{k+1})^i\right] \ .
\end{aligned}$$

By inspection it can be seen that the number of terms of degree j, $0 \leq j \leq k$ is given by

$$\binom{q + j - 1}{j} \ .$$

Thus the total number of terms, m, is just the summation of the above quantity from $j = 0$ to $j = k$. That is,

$$m = \sum_{j=0}^{k} \binom{q + j - 1}{j} \ .$$

Using the closed-form expression for the above summation, (see [GR80]), we obtain a general expression of the number of terms m in a k^{th}-degree polynomial in q variables.

$$m = \binom{k + q}{k} \ . \tag{E.1}$$

In particular, when the image is two dimensional, that is, $q = 2$ and the pixel coordinates can be expressed by the row and column, we have,

$$m = \frac{(k + 1)(k + 2)}{2} \ . \tag{E.2}$$

[25] The generating function in this case is $A(z)$.