Contents

Figures

Tables

Introduction

Computer vision refers to a variety of applications involving one or more cameras (or any other sensing devices) and computer algorithms for restoring or interpreting the image. The sensed data is transformed to an array of measured light intensities, each element corresponding to a small patch in the scene (a pixel). The obtained digitized image is the starting point of any kind of computer analysis. In some applications, the sensing device may be more specific responding to other form of light: infrared imaging, photon emission tomography, radar imaging, ultrasonic imaging, etc…

We can separate two levels in image processing: High level processing aims at extracting symbolic features (for example recognition of a character in a handwritten letter) while low level vision deals with extraction of image attributes needed for higher level processing.

The first step in almost every computer vision process, called *early vision*, involves a variety of digital image processing [103, 52] tasks dealing directly with massive amounts of pixel data. The goal of such a process is to transform the digitized image data into more meaningful tokens (edges, texture features, regions, etc…) for higher level processing.

Herein, we deal with a statistical approach of such early vision processes. In real scenes, neighboring pixels usually have similar intensities. In a probabilistic framework, such regularities are well expressed mathematically by Markov Random Fields. On the other hand, the local behavior of Markov Random Fields permits to develop highly parallel algorithms in the resolution of the combinatorial optimization problem associated with such a model. We also discuss parameter estimation methods in order to develop completely data-driven algorithms.

In Chapter 2, we propose a new hierarchical Markov Random Field model and in Chapter 4, we present a parameter estimation method for computing the model parameters. In Chapter 3, a new Multi-Temperature Annealing scheme is proposed and a rigorous mathematical study of the convergence is provided. We also propose a deterministic variant of the Metropolis algorithm, called Modified Metropolis Dynamics, with a detailed convergence study. All models and algorithms presented in this thesis have been tested on image segmentation problems. The algorithms have been implemented in parallel on a Connection Machine CM200 and the appropriate comparative tests are presented at the end of each chapter.

Image Processing

One of the first applications of image processing dates back to the early 20's when digitized images of news events were transmitted by a submarine cable between New York and London [52]. Pictures were coded to cable transmission and then decoded at the receiving end. The initial problem was related to improving the visual quality of these images.

With the construction of the third-generation digital computers in the mid 60's, applications of digital image processing become widespread. Since image processing usually require large speed and storage capabilities, it was the motivating application for the design of several parallel computers [71, 65].

The goal of computer vision is the processing of image data for autonomous machine perception. A computer vision system involves a sensing device (usually a camera) and computer algorithms to interpret the picture. The term *image* (more precisely, *monochrome image*) refers to a two dimensional light intensity function whose value at any point is proportional to the brightness (*grey-level*) of the image at that point [52]. A *digital image* is a discretized image both in spatial coordinates and in brightness. It is usually represented as a two dimensional matrix, the elements of such a digital array are called pixels.

Basically, there are two levels of image processing: *Low level* (or early vision) tasks deal with large amount of pixel data, transforming a digitized image into a form that can be used in *high level* processing. Herein, we are interested in early vision processes, in particular in the probabilistic modelization of such problems using Markov Random Fields.

Early Vision and Markov Random Fields

Early vision deals directly with raw pixel data involving [2] image compression, restoration [25, 46, 66, 115, 118, 114], edge detection [47, 108, 115, 118, 114], segmentation [42, 79, 70, 43, 34, 106, 107, 35, 60, 56, 116], texture analysis [28, 48, 61], motion detection [64], optical flow, etc... Most of these problems can be formulated in a general framework, called *image labeling*, where we associate a label to each pixel from a finite set. The meaning of this label depends on the problem that we are trying to solve. For image restoration, it means a grey-level; for edge detection, it means the presence or the direction of an edge; for image segmentation, it means a class (or region); etc... The problem here is how to chose a label for a pixel, which is *optimal* in a certain sense. Relaxation Labeling [67, 38] is a classical, non-probabilistic[1] solution of the problem.

Our approach is probabilistic: at each pixel, we want to select the most likely labeling. To achieve this goal, we need to define some probability measure on the set of all possible labelings. In real scenes, neighboring pixels have usually similar intensities; edges are smooth and often straight. In a probabilistic framework, such regularities are well expressed by Markov Random Fields (MRF). Another reason for dealing with MRF models is of course the *Hammersley-Clifford theorem* [14, 94] which allows to define MRF's through clique-potentials. In the labeling problem, this leads to the following Bayesian formulation: we are looking for the Maximum A Posteriori (MAP) estimate of the label field yielding to the minimization of a usually non-convex energy function.

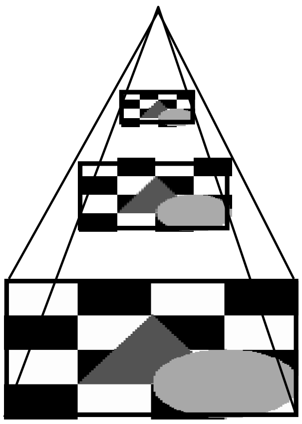

Figure 1: *Pyramidal representation.*

Unfortunately, finding such an estimate is a heavy computational problem. For example, if we consider a 16×16 image with only two possible labels at each pixel, we obtain a configuration space of 2^{256} elements. It is then impossible to find the optimum by computing the possible values of the cost function. On the other hand, due to the non-convexity, classical gradient descent methods cannot be used since they get stuck in a local minimum. In the early 80's, a Monte-Carlo algorithm, called Simulated Annealing, has been proposed independently by Černy [24] and Kirkpatrick *et al.* [82] to tackle the optimization. However, with the first substantial mathematical results [46, 58], it becomes clear that successful applications of Simulated Annealing (SA) require a very slow temperature cooling schedule and thus large computing time. To avoid this drawback, two solutions have been proposed: One of them deals with the possible parallelization of SA algorithms [5]. The other solution is to use *deterministic* algorithms, which are suboptimal, but converge within a few iterations requiring therefore less computing time [13, 79].

On the other hand, multigrid (or pyramidal) [17, 88, 98, 71] models can also significantly improve the convergence rate and the quality of the final result of iterative relaxation techniques. Multigrid methods have a long existence in numerical analysis (partial differential equations, for instance). In image processing, they have also been used in various context from the mid 70's [71]. Herein, we are interested in pyramidal methods applied to MRF image modelization. We use the word *pyramidal* to designate *multigrid* and *hierarchical* schemes. The essence of such an approach is to represent an image at multiple resolutions, arranging the coarse images in a pyramid as shown in

[1]There are probabilistic approaches of RL, but the original formulation in [67] is not probabilistic.

Figure 1.

We are talking about *multigrid* methods, if the layers in the pyramid are not connected. In this case, the optimization algorithm is usually parallelizable only on the layers, but it is still sequential between layers. The layout of a *multigrid* model can be represented by a stack of smaller and smaller image lattices. An important question in multigrid modelization is how to define the cliques and their potentials at coarse resolutions. There are various ideas [85] including the Renormalization Group approach of Gidas [50, 95], a consistent multiscale approach of Perez *et al.* [64, 63, 99], or Bouman's causal pyramidal model [16, 18].

If there is an inter-level communication, the model is called *hierarchical* [75, 74, 73]. While the optimization algorithms associated with such models can be parallelized on the whole pyramid, the underlying MRF model becomes more complicated requiring more computation.

Image Segmentation

Throughout this thesis, the proposed models and algorithms are tested on image segmentation (or image classification) problems. Herein, we briefly review the goals of such a processing.

First of all, let us define what are *image segmentation* and *classification*. Practically, they refer to the same task. However, we prefer to use the name *segmentation* because *classification* is a more general term. Classification [37] supposes a *feature extractor*, whose purpose is to reduce the data by measuring properties that distinguish picture elements (for example, the sea and the cities in Figure 2). Then the problem of classification is basically to partition the *feature space* into regions, one region for each category. Let us see an example. We have a satellite image shown on Figure 2, and we have to detect the urban areas. Using only grey-level values, we cannot distinguish between cities and agricultural areas since they are located in the same spectrum. The only feature which makes the difference between them is the *texture*. The feature space is then contains texture *and* grey-level values. A more complicated feature space could be obtained by considering geometrical properties, or it might involve the grey-levels of more than one channels (XS1, XS2, XS3 channels for SPOT images).

In our terminology, segmentation can be viewed as a special case of classification where the feature space only consists of grey-level values. The objective of segmentation is to partition an *image* into homogeneous regions. It may be done by finding boundaries between regions or by finding the regions directly without detecting edges between them. Herein, we are interested in the latter approach. The segmentation process aims at partitioning the image into regions such that [52]:

- The segmentation must be complete (i.e. every pixel must be in a region).
- The pixels in a region must be connected.
- The regions must be disjoint.

Classical segmentation processes can be found in [103, 52], such as *region growing algorithms* or *split and merge algorithms*.

Notice, that the second item in the above list means that nearby pixels must be in the same region. This constraint can be well expressed in terms of a Markov Random Field. To be more precise, let us associate a label to each pixel. This label simply means the type of region containing the pixel. Then, we define a MRF over these labels favoring similar labels at neighboring pixels by the definition of some potential functions. However, this would result in a segmentation where the entire image is a whole region. We need another constraint: a link between the observed grey-level values and the regions. The most natural model is to consider each class as a Gaussian distribution over the possible grey-levels. In this way, the regions are characterized by the mean value and the variance of the corresponding normal density function. On the other hand, we can introduce these distributions into the MRF model as the potential of the first order cliques (singletons). The resulting model is now able to correctly segment a grey-level image.

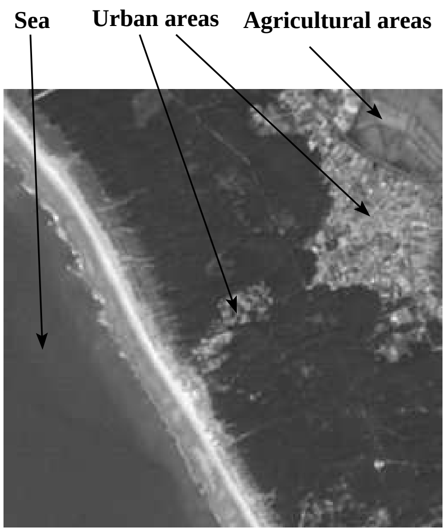

Figure 2: *A SPOT image.*

We remark that, in a Markovian framework, one could detect region boundaries and regions at the same time by introducing a *line process* [46] in the model. It consists of placing an edge element between two neighboring pixels (see [46, 115], for an example). However, we did not use such a process because our goal was not to establish a sophisticated, environment-specific model but to construct an easily applicable universal model and then study its multi-scale and hierarchical implementations.

Summary by Chapters

In Chapter 1, we discuss the basics of Markov Random Fields from a mathematical and physical point of view. We briefly describe the Ising and Potts models. We also present the Bayesian decision theory and define some standard notions such as probability, random variables, distribution, probability density function, probability mass function, different types of convergence of random variables, stochastic processes, etc... The Gaussian distribution is discussed in details because of its important role in statistical image analysis.

In Chapter 2, we establish Markovian image models in a general framework, called image labeling. We also present some related non-Markovian model such as relaxation labeling and the weak membrane model[2]. Of course, we discuss Bayesian estimation methods coupled with Markovian modelization. Then, we explain some multigrid techniques in order to reduce the computing time required by the MAP estimation and also to increase the quality of the obtained estimates. Finally, we propose a new hierarchical Markovian model which aims at incorporating cliques with far apart sites in order to get better estimates. While the computing time radically increases[3], the reached estimates are the best among the studied models, especially for deterministic relaxation algorithms.

In Chapter 3, we discuss combinatorial optimization algorithms. Since all of the presented models result in a non-convex energy function, the final result is strongly dependent on the effectiveness of the optimization algorithm. We explain the classical stochastic and deterministic relaxation algorithms with a detailed mathematical background and review some parallelization techniques. Then, we propose a new annealing scheme: the Multi-Temperature Annealing which exhibits faster convergence on hierarchical models than classical annealing schemes. The convergence towards a global optimum has been proved [73] with a generalization of the well known annealing theorem of Geman and Geman [46]. We also propose a new deterministic algorithm: Modified Metropolis Dynamics. It seems to be a good compromise between quality and rapidity. The mathematical study of the convergence towards a *local* minima is provided in a theorem.

[2] We notice that the weak membrane model has also been used in a Markovian context but originally, as proposed by Blake and Zisserman [15], it was a non-Markovian model

[3] Although, we believe that on a pyramid architecture better computing times could be achieved.

In Chapter 4, we review some parameter estimation methods in order to define a completely data-driven process. If we have a realization of the label field then the problem is relatively easy, we have many classical methods to do parameter estimation. Unfortunately, such a realization is not known in real life applications, so the direct use of such an estimation algorithm is impossible. We have to deal with iterative estimation methods consisting of subsequently generating a labeling, estimating parameters from it, then generating a new labeling using these parameters, etc . . . The initialization of such methods is very important because of its influence on subsequent labelings and hence on the final estimates. Since classes are mostly represented by a Gaussian distribution, we discuss in details a few methods to compute the modes of a Gaussian mixture without any a priori information. The algorithms have been applied to monogrid and hierarchical MRF models and tested on image segmentation problems.

In This Chapter:

1. Fundamentals

Herein, we discuss the principal fundaments of Markov Random Fields (MRF) from a mathematical and physical point of view. The physical one is very important because MRF theory was inspired by statistical mechanics (Ising model). In image processing, we use the same terminology as in statistical mechanics: energy, potential, temperature... Of course, the meaning of these terms differs from the ones used in statistical mechanics.

We are also interested in the theory of decision (mainly in the Bayesian sense) as the theoretical basis of the Maximum a Posteriori (MAP) estimates has widely been applied in image labeling. We define some standard notions such as probability, random variables, distribution, probability density function, probability mass function, different types of convergence of random variables, stochastic processes, etc... The Gaussian distribution is discussed in details because of its important role in statistical image analysis: It is the most broadly used tool to take into account the observations in a probabilistic model. The material in this chapter is mainly based on the book of Papoulis on the theory of probability [96].

1.1 Probability and Random Variables

We have various definitions of probability: one can define it as a ratio of favorable to total number of outcomes (classical) or in an axiomatic way (as a measure). We shall use the last one. However, we remark that the classical definition is a useful tool in the solution of physical problems. On the other hand, one can prove that this definition implicitly assumes that the axioms, which we will define in the next paragraph, are valid.

Let us denote by $\mathbf{I}$ the certain event (the one that occurs in every trial). Given two events A and B, $A \cup B$ denotes the event that occurs whenever A or B or both occur. A and B are mutually exclusive if the occurrence of one at a given trial excludes the other. Now, we can define the probability of an event by three postulates:

Definition 1.1.1 (Probability) *The probability of an event A is a number $P(A)$ assigned to this event satisfying the following axioms:*

(i) *$P(A)$ is positive: $P(A) \geq 0$*

(ii) *The probability of the certain event equals 1: $P(\mathbf{I}) = 1$*

(iii) *If A and B are mutually exclusive, then $P(A \cup B) = P(A) + P(B)$*

Now, let us examine what is an event. Since events may be combined in various ways to form other events, events are best described by the *set theory*. The outcomes of a single experiment are well defined objects (the elementary events) forming a set $\mathbf{I}$ called space. Events are various subsets of this space containing one or more elementary events. An event occurs when one of its elementary event occurs. Two events are mutually exclusive, if they have no common elements. It is clear that $\mathbf{I}$ is the certain event since it always occurs. The empty set $\emptyset$ corresponds to the impossible event which never occurs. Mathematically, we define an experiment in the following way:

Definition 1.1.2 (Borel Field) *$\mathcal{B}$ is a Borel Field if the sets $A_1, \ldots, A_n$ belongs to the field then*

$$\bigcup_{i=1}^{n} A_i \text{ and } \bigcap_{i=1}^{n} A_i \tag{1.1}$$

also belong to it.

Definition 1.1.3 (Experiment) *An experiment $\mathcal{E} = (\mathbf{I}, \mathcal{B}, P)$ is specified by the three following concepts:*

(i) A set $\mathbf{I}$ of outcomes called space or certain event.

(ii) A Borel field $\mathcal{B}$ consisting of certain subsets of $\mathbf{I}$ called events.

(iii) A number $P(A)$ is assigned to every event A, it is called probability satisfying the axioms in Definition 1.1.1.

In the following, we deal with random variables. Random variables are functions, satisfying some general conditions, which assign a number $X(\xi)$ to every outcome $\xi \in \mathbf{I}$. What does this number mean? It could be the gain or loss in a game, the length of a manufactured product, etc...

Example 1.1.1 Let us consider the rolling of a die and note the faces by $f_1, \ldots f_6$. Then, we could define the random variable $X(f_i) = 10i$ as our gain in a game of die.

Definition 1.1.4 (Random Variable) *A random variable X is a function whose domain is the space $\mathbf{I}$ assigning a number $X(\xi)$ to every outcome $\xi \in \mathbf{I}$ of the experiment $\mathcal{E} = (\mathbf{I}, \mathcal{B}, P)$ such that:*

(i) The set $\{X \leq x\}$ is an event for any number x or, in other words, X is measurable *in the field $\mathcal{B}$.*

(ii) The probability of the events $\{X = +\infty\}$ and $\{X = -\infty\}$ equals zero.

In the following, we define some functions to characterize a random variable:

Definition 1.1.5 (Distribution) *Given a random variable X, we call the function*

$$F_X(x) = P\{X \leq x\} \tag{1.2}$$

the distribution function *of X defined for any number $x \in (-\infty, \infty)$.*

Clearly, $F(-\infty) = 0$ and $F(\infty) = 1$.

Example 1.1.2 Let us consider again the rolling of a die with the random variable X

defined in Example 1.1.1. Since $P\{f_i\} = 1/6$, $F(x)$ is a staircase function with steps equals to $1/6$.

Definition 1.1.6 (Density) *Given a random variable X, we call the derivative of its distribution $F_X(x)$*

$$f(x) = \frac{dF(x)}{dx} \tag{1.3}$$

the density function *of X.*

From the monotonicity of $F(x)$ follows that $f(x) \geq 0$ and:

$$\int_{-\infty}^{\infty} f(x)dx = F(\infty) - F(-\infty) = 1 \tag{1.4}$$

Since $F(x)$ might not have a derivative for every x, we can't associate a density function to all random variables. For example, the distribution defined in Example 1.1.2 is not differentiable in the ordinary sense. However, we shall define a density even for this discrete type random variable by allowing $f(x)$ to contain impulses. $\delta(x)$ is an impulse function if

$$\int_{-\infty}^{\infty} \phi(x)\delta(x)dx = \phi(0) \tag{1.5}$$

holds for any function $\phi(x)$ continuous at 0. Shifting the origin, we get:

$$\int_{-\infty}^{\infty} \phi(x)\delta(x - x_0)dx = \phi(x_0) \tag{1.6}$$

Using the above equation, one can show that the derivative of a discontinuous function $F(x)$ at a discontinuity point x_0 equals to

$$(F(x_0^+) - F(x_0^-))\delta(x - x_0) \tag{1.7}$$

From this property follows that the density function of a discrete random variable consists of impulses at the points x_i:

$$f(x) = \frac{dF(x)}{dx} = \frac{d}{dx}\sum_i p_i = \sum_i p_i\delta(x - x_i) \tag{1.8}$$

The function $f(x)$ can be considered as a *probability mass*. If $f(x)$ is finite then the mass in the interval $(x, x + dx)$ equals $f(x)dx$. The impulses $p_i\delta(x - x_i)$ can be considered as *point masses* p_i placed at x_i. The *total mass* on the entire x axis equals 1 (see Equation (1.4)). The distribution $F(x)$ equals the mass in the interval $(-\infty, x)$.

Now, let us discuss one of the most important parameter of a random variable: the *expected value* or *mean*.

Definition 1.1.7 (Expected Value) *The expected value of a random variable X is the integral*

$$E\{X\} = \int_{-\infty}^{\infty} x f(x) dx \tag{1.9}$$

where $f(x)$ is the density of X.

We remark that a random variable not necessarily equals its expected value for any experimental outcome. If $f(x)$ is interpreted as a probability mass then the expected value is the center of gravity of this mass. If X is discrete then $E\{X\}$ is given by the sum $\sum x_n p_n$ which is expressed in terms of its values x_n and the probability $P\{X = x_n\}$. In the continuous case, we could express it by a Lebesgue integral:

$$E\{X\} = \int_{\mathbf{I}} x dP \tag{1.10}$$

Another important parameter is the *variance* giving some notion about the concentration of the density function near the expected value.

Definition 1.1.8 (Variance) *The variance of a random variable with mean μ is given by:*

$$\sigma^2 = E\{(X-\mu)^2\} = \int_{-\infty}^{\infty} (x-\mu)^2 f(x) dx \tag{1.11}$$

Its square root σ is called standard deviation.

From Equation (1.11), we obtain the important relationship

$$\sigma^2 = E\{X^2 - 2X\mu + \mu^2\} = E\{X^2\} - 2\mu E\{X\} + \mu^2 = E\{X^2\} - \mu^2 = E\{X^2\} - E^2\{X\} \tag{1.12}$$

To specify the statistics of a random variable, we can use its *moments*:

Definition 1.1.9 (Moments) *Given a random variable X, its moments m_k are given by:*

$$m_k = E\{X^k\} = \int_{-\infty}^{\infty} x^k f(x) dx$$

It is clear that $m_0 = 1$ and $m_1 = E\{X\}$.

Definition 1.1.10 (Central Moments) *The constant*

$$\eta_k = E\{(X-\mu)^k\} = \int_{-\infty}^{\infty} (x-\mu)^k f(x) dx \tag{1.13}$$

are called central moments.

We have $\eta_0 = 1$, $\eta_1 = 0$ and $\eta_2 = \sigma^2$. We can express the central moments in terms of the moments m_k:

$$\eta_k = \sum_{r=0}^{k} \binom{k}{r} (-1)^r \mu^r m_{k-r} \tag{1.14}$$

and similarly determine m_k from the central moments:

$$m_k = \sum_{r=0}^{k} \binom{k}{r} \mu^r \eta_{k-r} \tag{1.15}$$

We have also the *absolute moments* $M_k = E\{|X|^k\}$ and the *generalized moments* ${}_a m_k = E\{(X-a)^k\}$ and ${}_a M_k = E\{|X-a|^k\}$ with respect to a.

In the following, we consider two random variables and define their joint distribution and density functions. We remark that these definitions can be extended for an arbitrary number of random variables.

Definition 1.1.11 (Joint Distribution) *The joint distribution function of the random variable X and Y is defined by*

$$F_{XY}(x,y) = P\{X \leq x, Y \leq y\}.$$

The distributions $F_X(x)$ and $F_Y(y)$ are called marginals.

In general, the joint distribution cannot be determined from the marginals but it is related to them. Indeed,

$$F_{XY}(x,\infty) = F_X(x) \text{ and } F_{XY}(\infty,y) = F_Y(y) \tag{1.16}$$

since $\{y \leq \infty\}$ (resp. $\{x \leq \infty\}$) is the certain event.

Definition 1.1.12 (Joint Density) *Assuming that $F_{XY}(x,y)$ is partially differentiable up to order two, the joint density function is given by*

$$f_{XY}(x,y) = \frac{\partial^2 F_{XY}(x,y)}{\partial x \partial y} \tag{1.17}$$

As in the one dimensional case, we could define f(x,y) as a *probability mass* function, extending it to random variables whose joint distribution is not differentiable (see Equation (1.8)).

Definition 1.1.13 (Covariance) *The covariance of two random variables is defined by*

$$cov_{XY} = E\{(X-\mu_X)(Y-\mu_Y)\} \tag{1.18}$$

and the ratio

$$r = \frac{E\{(X-\mu_X)(Y-\mu_Y)\}}{\sqrt{E\{(X-\mu_X)^2\}E\{(Y-\mu_Y)^2\}}} = \frac{cov_{XY}}{\sigma_X \sigma_Y} \tag{1.19}$$

is called the correlation coefficient.

Two random variables X and Y are *independent* if the events $\{X \leq x\}$ and $\{Y \leq y\}$ are independent. This yields $F_{XY}(x,y) = F_X(x)F_Y(y)$ and $f_{XY}(x,y) = f_X(x)f_Y(y)$. Furthermore, X and Y are *uncorrelated* if $E\{XY\} = E\{X\}E\{Y\}$ and they are orthogonal if $E\{XY\} = 0$. Independence is a stronger condition than uncorrelatedness.

1.2 Assigning Probabilities

In Definition 1.1.1, we defined the probability in an axiomatic way as a measure. We said that the probability is a *number* assigned to an event satisfying some axioms. But we didn't tell *how* to assign these numbers! One possibility would be to use the classical definition which says that the probability of an event A is found by counting the total number N of the possible outcomes of the experiment. If N_A is the number of outcomes when A occurs then

$$P(A) = \frac{N_A}{N} \tag{1.20}$$

While this would be a reasonably good definition in physical applications such as the die experiment in Example 1.1.1, in statistical image processing, we will assign probabilities to propositions such as "The image X is a better description of the observations than Y" or "The image was affected by an additive Gaussian noise with parameters μ and σ" etc... On the other hand, if in the die example the rolling is not fair then the above definition is not usable since we implicitly assumed that the outcomes are equally likely. Thus, we need a definition which allows to incorporate information that one actually possesses. The *maximum entropy principle* is one of the possibilities. The entropy is a measure of the information content of (or the uncertainty about) a random variable:

Definition 1.2.1 (Entropy) *Given a discrete random variable X with distribution $P(x)$, its entropy is defined by*

$$H(X) = -\sum_x P(x) \log P(x) \tag{1.21}$$

This definition is often referred to as *Shannon's formula* since it was first developed by Shannon in information theory. A detailed work about information theory is [30]. The *maximum entropy principle* says that one can assign a probability to an event by maximizing the entropy subject to some constraints (ie. the information that we actually possess). In the following example, we will demonstrate this principle on the die experiment [20]:

Example 1.2.1 Suppose that nothing is known about the die, except that the probability distribution should be normalized:

$$1 - \sum_{i=1}^{6} p_i = 0$$

We can multiply the above constraint by c and add it to the entropy without changing the value of H since it equals 0:

$$H = -\sum_{i=1}^{6} p_i \log p_i + c\left(1 - \sum_{i=1}^{6} p_i\right)$$

To assign the probabilities p_i to the faces, H is maximized with respect to the unknowns p_i $(i = 1, \ldots, 6)$ and c. Differentiating H with respect to the variables, we obtain seven equations

$$\begin{aligned} -(\log p_i + 1) + c &= 0 \quad i = 1, \ldots 6 \\ 1 - \sum_{i=1}^{6} p_i &= 0 \end{aligned}$$

from which we get $p_i = 1/6$ and $c = 1 - \log 6$.

What we can see is that the maximum entropy principle reduces to the uniform prior if the only information is that the probabilities should total one. However, we could obtain a nonuniform distribution in the above example if information were available that the rolling of die is not fair. Another example of the maximum entropy principle can be found in [20] for parameter estimation.

1.3 Bayesian Probability Theory

While it is relatively easy to estimate the distribution of a random variable within an ensemble of data sets, this is not the problem faced in image processing. We have typically only one data set and we are trying to determine the parameters. To solve this problem a wider interpretation is needed within probability theory. This interpretation is called *Bayesian probability theory.* It is named after *Rev. Thomas Bayes*, an 18^{th} century mathematician. He derived the *Bayes theorem* which is the starting point for all Bayesian calculations.

The basic concept in Bayesian probability theory is that *all* probabilities are conditional. The use of $P(A)$ does not make sense until the evidence on which it is based is given. Nevertheless, we shall use this notation for brevity with the remark that it stands for $P(A|.)$ where '.' is the event upon which A is based.

Definition 1.3.1 (Conditional Probability) *Given an event C with nonzero probability, the conditional probability of A given C is defined by*

$$P(A|C) = \frac{P(A \cap C)}{P(C)} \tag{1.22}$$

Indeed, if A and C are mutually exclusive then $P(A|C) = 0$ and if $A \subset C$ then $P(A \cap C) = P(A)$. Given an event C, the conditional probabilities $P(.|C)$ satisfy the axioms in Definition 1.1.1. This fact enables us to define a new experiment:

Definition 1.3.2 (Bayesian Experiment) *Given an experiment $(\mathbf{I}, \mathcal{B}, P(A))$ and an event C with nonzero probability, we define a new experiment by $(\mathbf{I}, \mathcal{B}, P(A|C))$ which we call* Bayesian experiment.

This experiment has the same outcomes and the same events but a new probability measure.

There are two basic rules for manipulating probabilities: The *product rule* and the *sum rule*:

$$\text{Product rule:} \quad P(A, B|C) = P(A|C)P(B|A, C) \tag{1.23}$$

$$\text{Sum rule:} \quad P(A \cup B|C) = P(A|C) + P(B|C) - P(A, B|C) \tag{1.24}$$

The *sum rule* has an important role in the following theorem on total probability (For simplicity, we shall write $P(A)$ instead of $P(A|C)$). This theorem is used to evaluate $P(B)$ in terms of $P(B|A_i)$ and $P(A_i)$.

Theorem 1.3.1 (Total Probability) *Given n mutually exclusive events $A_1, \ldots, A_n$ whose sum is the certain event:*

$$A_i \cap A_j = \emptyset \quad \forall i \neq j, \quad i = 1, \ldots n$$

$$\bigcup_{i=1}^{n} A_i = \mathbf{I}$$

The following equation holds for any arbitrary event B:

$$P(B) = \sum_{i=1}^{n} P(B|A_i)P(A_i) \tag{1.25}$$

We remark that the above theorem is still valid even if the sum of the events A_i is not the certain event, provided that

$$\bigcup_{i=1}^{n} A_i \supset B.$$

Now, we present the Bayes theorem, which is the most important theorem in the Bayesian probability theory. As we mentioned earlier, it is the basis of all Bayesian calculations.

Theorem 1.3.2 (Bayes)

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

The probability $P(A|B)$ is called *a posteriori probability* and $P(A)$ is the *a priori probability* of A.

In the following, we define the *conditional distribution* and *density*. This is nothing else but the interpretation of the corresponding definitions in the *Bayesian experiment* $(\mathbf{I}, \mathcal{B}, P(A|C))$. As an example, we redefine the distribution:

Definition 1.3.3 (Conditional Distribution) *Given an event C with nonzero probability, the conditional distribution of a random variable X is given by*

$$F_X(x|C) = P\{X \leq x|C\} = \frac{P\{X \leq x, C\}}{P(C)} \tag{1.26}$$

The *conditional density* is the derivative of the distribution as in the classical case.

Let us consider now the condition C. If it is expressed in terms of the random variable X (suppose $C = \{b < X \leq a\}$) then the above distribution can be given in terms of $F_X(x)$:

$$F(x|b < X \leq a) = \begin{cases} 1 & \text{if } x \geq a \\ \frac{F(x)}{F(a)} & \text{if } b \leq x < a \\ 0 & \text{if } x < b \end{cases} \tag{1.27}$$

The total probability can also be expressed in terms of the distribution:

$$F(x) = \sum_{i=1}^{n} F(x|A_i)P(A_i) \tag{1.28}$$

The conditional probability $P(A|C)$ is undefined if $P(C) = 0$. If C is expressed as a random variable, namely $C = \{X = x\}$ then one can define $P(A|C)$ as the following limit assuming that $f(x) \neq 0$:

$$P(A|X = x) = \lim_{\Delta x \to 0} P(A|x < X \leq x + \Delta x) = \frac{f(x|A)P(A)}{f(x)} \tag{1.29}$$

From the above equation, we can deduce the *continuous version* of the total probability theorem (Theorem 1.3.1)

$$P(A) = \int_{-\infty}^{\infty} P(A|X = x)f(x)dx \tag{1.30}$$

and the Bayes theorem (Theorem 1.3.2)

$$f(x|A) = \frac{P(A|X = x)f(x)}{\int_{-\infty}^{\infty} P(A|X = x)f(x)dx} \tag{1.31}$$

The conditional moments are defined as in Section 1.1 but the density function is replaced by the conditional density. We define the conditional mean as an example:

Definition 1.3.4 (Conditional Expected Value) *The conditional expected value of a random variable X assuming C is given by*

$$E\{X|C\} = \begin{cases} \int_{-\infty}^{\infty} xf(x|C)dx & \textit{for the continuous case} \\ \sum_n x_n P(X = x_n|C) & \textit{for the discrete case} \end{cases} \tag{1.32}$$

Now, we deal with two random variables and define their conditional distribution, density and expected value. We defined the conditional distribution of a random variable in Definition 1.3.3. In the following, we examine this definition in the case where the condition C is expressed in terms of another random variable. First, let us assume that $C = \{x_1 < X \leq x_2\}$. In this case $P(C) = F_X(x_2) - F_X(x_1)$ and we have

$$F_Y(y|x_1 < X \leq x_2) = \frac{P\{x_1 < X \leq x_2, Y \leq y\}}{P\{x_1 < X \leq x_2\}} = \frac{F_{XY}(x_2, y) - F_{XY}(x_1, y)}{F_X(x_2) - F_X(x_1)} \tag{1.33}$$

Differentiating $F_Y(y|x_1 < X \leq x_2)$, the density is of the following form

$$f_Y(y|x_1 < X \leq x_2) = \frac{\int_{x_1}^{x_2} f_{XY}(x, y)dx}{F_X(x_2) - F_X(x_1)} \tag{1.34}$$

Next, we discuss a single point $C = \{X = x\}$ and, using Equation (1.33), define the conditional distribution as the limit:

$$\begin{aligned} F_Y(y|X = x) &= \lim_{\Delta x \to 0} F_Y(y|x < X \leq x + \Delta x) && (1.35) \\ &= \lim_{\Delta x \to 0} \frac{F_{XY}(x + \Delta x, y) - F_{XY}(x, y)}{F_X(x + \Delta x) - F_X(x)} && (1.36) \\ &= \frac{\partial F_{XY}(x, y)/\partial x}{dF_X(x)/dx} && (1.37) \end{aligned}$$

Finally, we give the distribution if C is specified in terms of both random variables ($C = \{X \leq a, Y \leq b\}$):

$$F_Y(y|X \leq a, Y \leq b) = \begin{cases} 1 & \text{if } y \geq b \\ \frac{F_{XY}(a,y)}{F_{XY}(a,b)} & \text{if } y < b \end{cases} \tag{1.38}$$

One can also define *joint conditional distributions*. For example, let $C = \{a < X \leq b\}$:

$$F_{XY}(x, y|a < X \leq b) = \begin{cases} \frac{F_{XY}(b,y) - F_{XY}(a,y)}{F_X(b) - F_X(a)} & \text{if } x > b \\ \frac{F_{XY}(x,y) - F_{XY}(a,y)}{F_X(b) - F_X(a)} & \text{if } a < x \leq b \\ 0 & \text{if } x \leq a \end{cases} \tag{1.39}$$

An important concept is the *conditional independence*.

Definition 1.3.5 (Conditional Independence) *We say that X_1 is conditionally independent of X_2, assuming X_3, if*

$$f(x_1, x_2|x_3) = f(x_1|x_3)f(x_2|x_3) \tag{1.40}$$

Independence and conditional independence does not imply each other. Clearly, from the above equation, does not follow that $f(x_1, x_2) = f(x_1)f(x_2)$ and from independence does not follow the above equation.

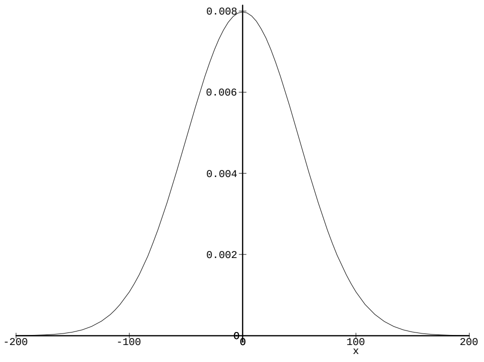

Figure 1.1: *Density function of a normal random variable.*

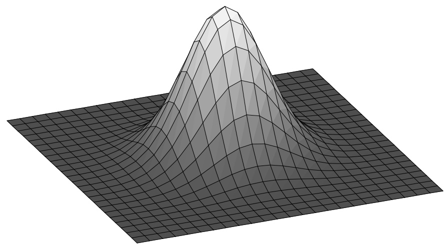

Figure 1.2: *Joint density function of two normal random variables.*

1.4 Normal Distribution

Normal or Gaussian distribution is the most frequently used probability distribution in image processing. For example, it is a standard assumption about a noise that it follows a Gaussian distribution.

Definition 1.4.1 (Normal Distribution) *We say that a random variable is normally distributed if its density function is a Gaussian curve (see Figure 1.1)*

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{1.41}$$

where μ is the mean (Definition 1.1.7) and σ is the standard deviation (Definition 1.1.8).

The distribution function is given by the integral of the above density:

$$\begin{aligned} F(x) &= \int_{-\infty}^{x} f(x)dx = \int_0^{\infty} f(x)dx + \int_0^x f(x)dx && (1.42) \\ &= \frac{1}{2} + \text{erf}\left(\frac{x-\mu}{\sigma}\right) && (1.43) \\ \text{with} \qquad & \text{erf}(x) = \frac{1}{\sqrt{2\pi}} \int_0^x \exp\left(-\frac{t^2}{2}\right) dt && (1.44) \end{aligned}$$

Now, let us consider two random variables.

Definition 1.4.2 (Jointly Normal Distribution) *Two random variables X and Y are jointly normal, if their density is of the form (see Figure 1.2)*

$$f(x,y) = \frac{1}{2\pi\sigma_X\sigma_Y\sqrt{1-r^2}} \exp\left(- \frac{\left(\frac{(x-\mu_X)^2}{\sigma_X^2} - \frac{2r(x-\mu_X)(y-\mu_Y)}{\sigma_X\sigma_Y} + \frac{(y-\mu_Y)^2}{\sigma_Y^2} \right)}{2(1-r^2)} \right). \quad (1.45)$$

where μ_X, μ_Y are the means and σ_X, σ_Y are the standard deviations of X and Y respectively. r is the correlation coefficient (see Definition 1.1.13).

One can show that if two random variables are jointly normal then they are *marginally normal*. Obviously, if they are independent then the converse is true. However, it is not necessarily true if they are not independent!

In the n-dimensional case, we can simply express the joint distribution by the covariance matrix Σ and the mean vector $\vec{\mu}$.

$$f(x_1,\ldots,x_n) = f(\vec{x}) = \frac{1}{\sqrt{(2\pi)^n|\Sigma|}} \exp\left(-\frac{1}{2}(\vec{x}-\vec{\mu})\Sigma^{-1}(\vec{x}-\vec{\mu})^T \right) \quad (1.46)$$

where

$$\Sigma = \begin{pmatrix} \sigma_{11} & \ldots & \sigma_{1n} \\ \ldots & \ldots & \ldots \\ \sigma_{n1} & \ldots & \sigma_{nn} \end{pmatrix}, \quad \mu = \begin{pmatrix} \mu_1 \\ \vdots \\ \mu_n \end{pmatrix} \quad (1.47)$$

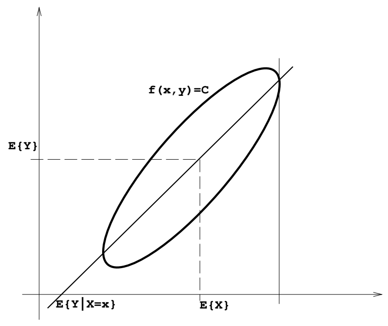

Figure 1.3: *Set of points where $f(x,y)$ is constant.*

If the random variables X_i are uncorrelated then their covariance matrix is diagonal and the joint density function can be factorized:

$$f(x_1,\ldots,x_n) = f(x_1)\cdots f(x_n) \quad (1.48)$$

Furthermore, we remark that if X_1 and X_2 are independent of X_3 and they are normal then the group (X_1, X_2) is also independent of X_3:

$$\left. \begin{array}{c} f(x_1,x_3) = f(x_1)f(x_3) \\ \text{and} \\ f(x_2,x_3) = f(x_2)f(x_3) \end{array} \right\} \text{then}$$

$$f(x_1,x_2,x_3) = f(x_1,x_2)f(x_3) \quad (1.49)$$

For the parameter estimation methods discussed in Chapter 4, it will be useful to examine the graph of a jointly normal density function from a geometrical point of view: The focus of points in the XY plane such that $f(x,y)$ is constant is given by the following equation

$$\frac{(x-\mu_X)^2}{\sigma_X^2} - \frac{2r(x-\mu_X)(y-\mu_Y)}{\sigma_X\sigma_Y} + \frac{(y-\mu_Y)^2}{\sigma_Y^2} = C \tag{1.50}$$

which defines an ellipse with center (μ_X, μ_Y) (see Figure 1.3). The density has the maximum at this point. Moreover, $E\{Y|X=x\}$ is a straight line passing through this center.

The conditional density of Y assuming $X=x$ can be derived from Equation (1.45):

$$f_Y(y|X=x) = \frac{f_{XY}(x,y)}{f_X(x)} = \frac{1}{\sigma_Y\sqrt{2\pi(1-r^2)}}\exp\left(-\frac{\left(y-\mu_Y-\frac{r\sigma_Y}{\sigma_X}(x-\mu_X)\right)^2}{2\sigma_Y^2(1-r^2)}\right) \tag{1.51}$$

which is also normal with mean and variance given by

$$E\{y|X=x\} = \mu_Y + \frac{r\sigma_Y}{\sigma_X}(x-\mu_X), \qquad \sigma = \sigma_Y^2(1-r^2)$$

In the following, we discuss some recursion formulas to compute various kind of moments of normal random variables. These formulas are used in a parameter estimation algorithm called *method of moments* (cf. Chapter 4). The next formula gives the moments of a zero-mean normal random variable:

$$E\{X^n\} = \begin{cases} 1\cdot 3\cdots(n-1)\sigma^n & \text{for } n \text{ even} \\ 0 & \text{for } n \text{ odd} \end{cases} \tag{1.52}$$

$$E\{|X|^n\} = \begin{cases} 1\cdot 3\cdots(n-1)\sigma^n & \text{for } n = 2k \\ \sqrt{\frac{2}{\pi}}2^k k!\sigma^{2k+1} & \text{for } n \text{ odd} \end{cases} \tag{1.53}$$

In the general case, we have a recursion formula as a function of its mean μ and variance σ^2:

$$m_k = E\{X^n\} = \frac{k(k-1)}{2}\int_0^{\sigma^2} m_{k-2}\,d\sigma^2 + \mu^k \text{ with } m_0 = 1 \text{ and } m_1 = \mu \tag{1.54}$$

For the central moments η_k, we can deduce a similar formula:

$$\eta_k = E\{(x-\mu)^k\} = \frac{k(k-1)}{2}\int_0^{\sigma^2} \eta_{k-2}\,d\sigma^2 \text{ with } \eta_0 = 1 \text{ and } \eta_1 = 0 \tag{1.55}$$

The *joint moments* of two jointly normal random variables with covariance ς (see Definition 1.1.13) can also be obtained by a recursion formula:

$$E\{X^kY^l\} = kl\int_0^{\varsigma} E\{X^{k-1}Y^{l-1}\}\,d\varsigma + E\{X^k\}E\{Y^l\} \tag{1.56}$$

Finally, one can prove the following useful relationships:

$$\begin{aligned} E\{XY\} &= r\sigma_X\sigma_Y && (1.57)\\ E\{X^2Y^2\} &= E\{X^2\}E\{Y^2\} + 2E^2\{XY\} && (1.58)\\ E\{|XY|\} &= \frac{2\sigma_X\sigma_Y}{\pi}(\cos(\alpha) + \alpha\sin(\alpha)) && (1.59)\\ \text{where} \quad & \sin(\alpha) = r \quad -\frac{\pi}{2} < \alpha \le \frac{\pi}{2} && (1.60) \end{aligned}$$

Theorem 1.4.1 *If two jointly normal random variable are uncorrelated ($r = 0$), then they are independent.*

1.4.1 White Noise

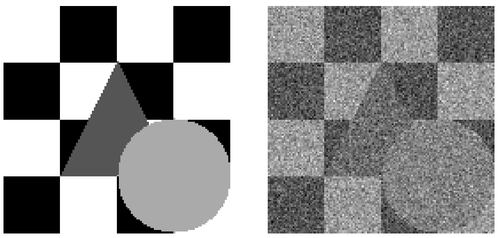

Figure 1.4: *Noisy image (3dB).*

A useful noise model in image processing is the *white noise* model [68]. A white random sequence $\{X_1, X_2, \ldots\}$ is a Markov sequence[1] with

$$P(X_k \mid X_l, l < k) = P(X_k). \tag{1.61}$$

That is, they are mutually independent which means that a white noise is completely random. If we assume that the X_k's are normally distributed, the sequence $\{X_1, X_2, \ldots\}$ is called a *white Gaussian random sequence.* In practice, we usually deal with such noises. The probability density of the sequence is determined by the mean vector $\vec{\mu}$ and the covariance matrix Σ:

$$\begin{aligned} \forall n \ge 1: \quad \vec{\mu}_n &= E\{X_n\} && (1.62)\\ \forall n, m \ge 1: \quad \Sigma_{nm} &= E\{(X_n - E\{X_n\})(X_m - E\{X_m\})^T\}. && (1.63) \end{aligned}$$

Since the sequence is white, the covariance matrix is diagonal and positive semidefinite. The most often used noise model is a *zero mean white Gaussian noise with deviation* σ with the following representation:

$$\vec{\mu} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \text{ and } \Sigma = \begin{pmatrix} \sigma^2 & & \mathbf{0} \\ & \sigma^2 & \\ \mathbf{0} & & \ddots \\ & & & \sigma^2 \end{pmatrix}. \tag{1.64}$$

[1] Markov processes will be discussed later in Section 1.7.

A noise is generally characterized by the *Signal to Noise Ratio* (SNR). The SNR is measured in dB according to the following formula:

$$\text{SNR in } dB \ = 10 \lg \left(\frac{\sigma^2_{image}}{\sigma^2_{noise}} \right), \tag{1.65}$$

where σ_{image} is the variance of the image. In Figure 1.4, we show an image corrupted by a $3dB$ zero mean white Gaussian noise.

1.5 Convergence and the Law of Large Numbers

We say, that a sequence of numbers x_n tends to a limit x if for any $\epsilon > 0$, there exists an integer n_0 such that

$$\forall n > n_0 : \quad |x_n - x| < \epsilon \tag{1.66}$$

The above (analytical) meaning of convergence is too restrictive in many cases in the theory of probability. This is why we shall redefine this notion in some weaker sense:

Definition 1.5.1 (Convergence with Probability 1) *The sequence X_n converges to X with probability 1 if the set of outcomes ξ such that*

$$\lim_{n \to \infty} X_n(\xi) = X(\xi) \tag{1.67}$$

has a probability equal to 1. This is written in the form

$$P\{X_n \to X\} = 1 \textit{ for } n \to \infty \tag{1.68}$$

Definition 1.5.2 (Convergence in the Mean-Square Sense) *The sequence X_n tends to X in the mean-square sense if*

$$\lim_{n \to \infty} E\{|X_n - X|^2\} = 0 \tag{1.69}$$

Definition 1.5.3 (Convergence in Probability) *Consider the probability that $|X_n - X| > \epsilon$, given the number $\epsilon > 0$: $P\{|X_n - X| > \epsilon\}$. If it converges to zero for every ϵ,*

$$\forall \epsilon > 0 : \quad \lim_{n \to \infty} P\{|X_n - X| > \epsilon\} = 0 \tag{1.70}$$

then the sequence X_n tends to X in probability.

Definition 1.5.4 (Convergence in Distribution) *Let $F_n(x)$ and $F(x)$ denote the distribution of the random variables X_n and X, respectively. If*

$$\lim_{n\to\infty} F_n(x) = F(x) \tag{1.71}$$

for every point x in which $F(x)$ is continuous, then X_n converges to X in distribution.

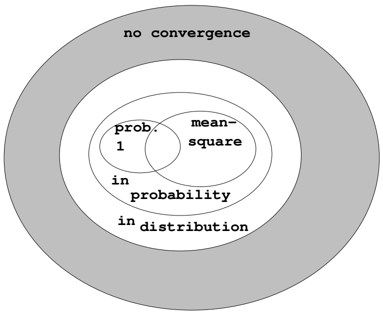

Figure 1.5: *Relation between different convergence modes.*

As we can see, we have various definitions of convergence in a more or less weak sense. Figure 1.5 shows the relation between them. For example, if X_n converges to X in the mean-square sense then it converges also in probability. If the limit X is not known, one can use the *Cauchy criterion* to test convergence:

$$\forall m > 0 : \quad \lim_{n\to\infty} |X_n - X_{n+m}| = 0. \tag{1.72}$$

If the above limit exists in one of the four sense defined earlier then X_n converges in the same sense.

An important theorem in statistics is the law of large numbers:

Theorem 1.5.1 (Law of Large Numbers) *If the probability of an event A in a given experiment is p and the experiment is repeated n times, then for any $\epsilon > 0$,*

$$\lim_{n\to\infty} P\{|\frac{k}{n} - p| \leq \epsilon\} = 1 \tag{1.73}$$

where k equals the number of successes of A

Borel showed a stronger statement, namely that it converges not only in probability but also with probability 1. Another related theorem is the *central-limit* theorem. Considering a sequence X_n of independent random variables with densities $f_i(x)$, the law of large numbers says that the variance of $\bar{X} = \sum_i X_i/n$ is small for large n. The essence of the *central-limit* theorem is that as n increases, $f_{\bar{X}}(x)$ tends to a normal (Gaussian) curve regardless of the shape of the densities $f_i(x)$.

1.6 Decision Theory

Decision theory is an approach to treat problems of mathematical statistics (see [39] for details on the subject). Not all parts of statistical theory could be justified from this view-point. For instance, maximum likelihood estimates (widely used in parameter estimation of statistical image models) are based on large sample theory. Decision theory is strongly related to the *theory of games* founded by John von Neumann[2], one of the outstanding mathematicians of the 20^{th} century. Decision theory may be considered as a theory which takes the role of one of the players in a two-person game. Thus, let us first define a *game*:

Definition 1.6.1 (Game) *A zero-sum two-person game consists of the following three elements:*

(i) Θ *– a nonempty set of possible states. Sometimes referred to as the parameter space.*

(ii) $\mathcal{A}$ *– a nonempty set of actions available.*

(iii) $L(\vartheta, a)$ *– a real valued loss function defined on* $\Theta \times \mathcal{A}$.

A game is just such a triplet $(\Theta, \mathcal{A}, L)$.

We suppose that before making a decision, we can look at the observed value of a (one or more dimensional) random variable X. The distribution of this random variable depends on the true state $\vartheta \in \Theta$. Using the notions of Section 1.1, X is defined on the Borel field $\mathcal{B}$ of $\mathcal{X}$ ($\mathcal{X}$ is the *sample space*, and for each $\vartheta \in \Theta$ there is a probability measure $P(.|\vartheta)$ on $\mathcal{B}$). The corresponding distribution function is denoted by $F_X(x|\vartheta)$. A *statistical decision problem* is nothing else but a *game* coupled with an observable random variable whose distribution depends on the true state $\vartheta \in \Theta$: Given the observed value of X is x ($X = x$), one chooses an action $d(x) \in \mathcal{A}$. Such a function d is an elementary strategy in this situation. The loss is the random quantity $L(\vartheta, d(X))$.

Definition 1.6.2 (Risk Function) *The expected value of* $L(\vartheta, d(X))$*, when* ϑ *is the true state is called the risk function*

$$R(\vartheta, d) = E\{L(\vartheta, d(X))|\vartheta\} = \int L(\vartheta, d(X))dP(x|\vartheta) \tag{1.74}$$

[2] His original name was Neumann János. He was born in Hungary and emigrated to the USA before the Second World War.

Definition 1.6.3 (Decision Rule) *The function $d(x) : \mathcal{X} \longrightarrow \mathcal{A}$ is a non-randomized decision rule if the risk function is finite for all $\vartheta \in \Theta$.*

In the following example, we define the risk function and the possible decision rules in an "odd or even" game [39].

Example 1.6.1 Two players simultaneously put up either one or two fingers. Player 1 wins if the sum is odd and player 2 wins if it is even. In all cases, the winner receives the sum of the digits showed (see Table 1.1 for the loss function of player 2). Now, let us consider the

$\Theta \setminus \mathcal{A}$	1	2
1	-2	3
2	3	-4

Table 1.1: *Loss function of the "odd or even" game.*

Θ	d_1	d_2	d_3	d_4
1	-2	$-3/4$	$7/4$	3
2	3	$-9/4$	$5/4$	-4

Table 1.2: *Risk function of the "odd or even" game.*

strategy of this player (remember, that decision theory may be considered as a theory which takes the role of one of the players). Suppose that before putting up the fingers, player 2 is allowed to ask player 1 how many fingers he intends to put up and that player 1 answers truthfully with probability 3/4. Therefore player 2 observes a random variable X taking the values 1 or 2. Indeed, $P(X = 1|\vartheta = 1) = 3/4$ and $P(X = 1|\vartheta = 2) = 1/4$ (ϑ means the number showed by player 1, that is the true state). Player 2 has four possible functions (*decision rules*) from $\mathcal{X} = \{1, 2\}$ into $\mathcal{A} = \{1, 2\}$:

$$\begin{array}{ll} d_1(1) = 1, & d_1(2) = 1; \\ d_2(1) = 1, & d_2(2) = 2; \\ d_3(1) = 2, & d_3(2) = 1; \\ d_4(1) = 2, & d_4(2) = 2; \end{array}$$

d_1 and d_4 ignore the value of X, d_2 guesses that player 1 is telling truth, and d_3 that he is not telling truth. The *risk function* is given by Table 1.2.

It is not always a good idea to use a deterministic rule. Sometimes it is more convenient to choose an action with a certain probability (depending on the observed value of X, for example). Such a decision rule is called *randomized decision*. Let Y be a random variable over $\mathcal{A}$ with distribution $F_Y(y)$. The expected loss associated with $F_Y(y)$ is then given by:

$$L(\vartheta, F_Y(y)) = E\{L(\vartheta, Y)\} \tag{1.75}$$

We have various types of *randomized decision rules*. The most interesting one for us is the so called *behavioral decision rule*. It is a function δ which gives for each x in the sample space a probability distribution over $\mathcal{A}$.

Definition 1.6.4 (Behavioral Risk Function) *Given a behavioral decision rule δ, the risk function is defined as*

$$\hat{R}(\vartheta, \delta) = E\{L(\vartheta, \delta(X))|\vartheta\}, \tag{1.76}$$

where L is the loss function defined by Equation (1.75).

Definition 1.6.5 (Behavioral Decision Rule) *The function $\delta(x): \ \mathcal{X} \longrightarrow \mathcal{A}^*$ is called behavioral decision rule if the risk function is finite for all $\vartheta \in \Theta$. $\mathcal{A}^*$ denotes the space of randomized decisions, δ, for which $L(\vartheta, \delta)$ is finite for all $\vartheta \in \Theta$.*

The fundamental problem of decision theory is to find optimal decision rules. Since a *best* rule usually does not exists, methods are needed for arriving at a *reasonable* decision rule. Here, we deal with the *Bayesian principle*, but other methods exist like *unbiasedness* or the *minimax principle*. Bayesian principle leads to an ordering of the available decision rules. First, we need a *prior distribution* on the parameter space Θ (sometimes referred to as *world model*).

Definition 1.6.6 (Bayes Risk) *The Bayes risk of a decision rule δ with respect to a prior distribution P is given by*

$$r(P, \delta) = E\{R(Y, \delta)\}, \tag{1.77}$$

where Y is a random variable over Θ with distribution P.

We also need the joint and conditional distribution of Y and X. The conditional distribution is also called the posterior distribution of the parameter given the observations. In using the Bayes principle, the parameter space is considered as a random variable whose distribution is known. For a fixed distribution, a rule is prefered to another one if it has a smaller Bayes risk. This relation sets up an ordering. The best rule with respect to this ordering is called a Bayes decision rule.

Definition 1.6.7 (Bayes Decision Rule) *Given a prior distribution P, δ_0 is a Bayes decision rule with respect to P if*

$$r(P, \delta_0) = \inf_{\delta} r(P, \delta). \tag{1.78}$$

The value $r(P, \delta_0)$ is known as the minimum Bayes risk.

If a Bayes rule does not exist, one must be satisfied with a rule whose risk is close to the minimum. The basic theorem of decision theory states that every really good decision rule is practically Bayesian. This is why Bayes procedures are the most broadly used tools.

1.7 Stochastic Processes and Markov Chains

First, we deal with stochastic processes[3] defining basic notions to the study of Markov chains. Markov chains will be extensively used in Chapter 3 in the convergence study of Simulated Annealing. Let us begin with the definition of a *stochastic process*.

Definition 1.7.1 (Stochastic Process) *We are given an experiment* $\mathcal{E} = (\mathbf{I}, \mathcal{B}, P)$. *To every outcome* $\xi \in \mathbf{I}$, *we assign a time function* $X(t, \xi)$. *Thus, we have created a function for each outcome* $\xi \in \mathbf{I}$. *The family of these functions is called a stochastic process. The notation* $X(t)$ *is also used.*

We remark that if t is fixed then $X(t, \xi)$ is a random variable. If these random variables are discrete (resp. continuous) then we say that the process has a *discrete* (resp. *continuous*) *state space*. If the time parameter t is discrete (resp. continuous) then the process is called *discrete* (resp. *continuous*) *parameter process*. If we regard $X(t)$ as a random variable, it is clear that its distribution will depend on t: $F(x;t) = P\{X(t) \leq x\}$. This function is called *first order distribution* of the process $X(t)$. Higher order distributions are defined similarly by the joint distributions of the random variables $X(t_1), \ldots, X(t_n)$:

$$F(x_1, \ldots, x_n; t_1, \ldots, t_n) = P\{X(t_1) \leq x_1, \ldots, X(t_n) \leq x_n\} \tag{1.79}$$

The corresponding n^{th}-order density of $X(t)$ is obtained by differentiating the above distribution. We can define in the same way the mean $\mu(t)$ of a stochastic process as the expected value of the random variable $X(t)$ (see Definition 1.1.7). The *autocorrelation* of a process is the joint moments of the random variables $X(t_1)$ and $X(t_2)$:

$$R(t_1, t_2) = E\{X(t_1)X(t_2)\} \tag{1.80}$$

The *auto-covariance* is the covariance of $X(t_1)$ and $X(t_2)$:

$$C(t_1, t_2) = E\{(X(t_1) - \mu(t_1))(X(t_2) - \mu(t_2))\} \tag{1.81}$$

[3]The terms *stochastic processes* and *random processes* are synonyms, the former is used generally when a time parameter is present.

Combining the above equations, we get:

$$C(t_1, t_2) = R(t_1, t_2) - \mu(t_1)\mu(t_2) \tag{1.82}$$

Definition 1.7.2 (Strict Stationarity) *A stochastic process is strictly stationary if its transition probabilities are not affected by a shift in the time. In other words, the processes $X(t)$ and $X(t+\epsilon)$ have the same statistics for any ϵ.*

Indeed, if $X(t)$ is strictly stationary then its n^{th}-order density must have the following form:

$$\forall \epsilon, \forall n: \quad f(x_1, \ldots, x_n; t_1, \ldots, t_n) = f(x_1, \ldots, x_n; t_1 + \epsilon, \ldots, t_n + \epsilon) \tag{1.83}$$

If the above equation holds only for $n \leq k$ then the process is said *stationary of order* k. As a special case of Equation (1.83), the first order density $f(x,t)$ is independent of t and consequently the expected value $E\{X(t)\}$ is constant. For the second order density, one concludes that it is a function of $r = t_1 - t_2$ and the autocorrelation depends only on the parameter r.

Definition 1.7.3 (Weak Stationarity) *A stochastic process is weakly stationary if its expected value is constant and its autocorrelation depends only on $r = t_1 - t_2$:*

$$E\{X(t)\} = \mu, \quad E\{X(t+r)X(t)\} = R(r) \tag{1.84}$$

1.7.1 Markov Chains

Markov property states that the future of a process depends only on the current state regardless how the process arrived at the given state. As a classical example of Markov processes, we mention the random walk.

Example 1.7.1 Considering the tossing of a fair coin, we take a step to the right if heads shows, to the left if tails shows. $X(t)$ denotes our position after t tossing. Indeed, $X(t)$ is a *stochastic process* satisfying the Markov principle.

Definition 1.7.4 (Markov Process) *A stochastic process $X(t)$ is called Markov process if for every n and $t_1 < t_2 < \cdots < t_n$ we have*

$$P\{X(t_n) \leq x_n | X(t_{n-1}), \ldots, X(t_1)\} = P\{X(t_n) \leq x_n | X(t_{n-1})\} \tag{1.85}$$

Let $X(t)$ be a continuous random variable and let $f(x|x_0)$ $t \geq t_0$ denote its conditional density given $X(t_0) = x_0$. If $t > t_1 > t_0$ then we have the following equation called *Chapman-Kolmogorov equation*:

$$f(x|x_0) = \int_{-\infty}^{\infty} f(x|x_1) f(x_1|x_0) dx_1 \tag{1.86}$$

Now, we discuss an important special case of Markov processes, the so-called *Markov chains.*

Definition 1.7.5 (Markov Chain) *A discrete parameter Markov process with discrete state space is called a Markov chain. In other words, let* $\{X_i\} = X_1, X_2, \ldots, X_n, \ldots$ *be a sequence of random variables taking the values* $a_1, \ldots, a_N$. *If the Markov property is satisfied:*

$$P\{X_n = a_{i_n} | X_{n-1} = a_{i_{n-1}}, \ldots, X_1 = a_{i_1}\} = P\{X_n = a_{i_n} | X_{n-1} = a_{i_{n-1}}\} \tag{1.87}$$

then $\{X_i\}$ *is a Markov chain.*

Let us denote the unconditional and conditional probabilities in the following way:

$$p_i(n) = P\{X_n = a_i\} \tag{1.88}$$

$$P_{ij}(n, s) = P\{X_n = a_i | X_s = a_j\} \tag{1.89}$$

The conditional probabilities $P_{ij}(n, s)$ are also called the *probabilities of transition.*The following equations are obvious:

$$p_i(n) = \sum_{j=1}^{N} P_{ij}(n, s) p_j(s) \tag{1.90}$$

$$\sum_{i=1}^{N} p_i(n) = 1 \tag{1.91}$$

$$\sum_{i=1}^{N} P_{ij}(n, s) = 1 \tag{1.92}$$

The discrete *Chapman-Kolmogorov equation* is of the form:

$$P_{ij}(n, s) = \sum_{k=1}^{N} P_{ik}(n, r) P_{kj}(r, s) \tag{1.93}$$

The transition probabilities $P_{ij}(n, s)$ could be arranged in a matrix and the unconditional probabilities $p_i(n)$ in a vector:

$$P(n, s) = \begin{pmatrix} P_{11}(n, s) & P_{12}(n, s) & \ldots \\ P_{21}(n, s) & P_{22}(n, s) & \ldots \\ \vdots & \vdots & \ddots \end{pmatrix}, \quad p_i(n) = \begin{pmatrix} p_1(n) \\ p_2(n) \\ \vdots \end{pmatrix} \tag{1.94}$$

In this context, Equation (1.92) means that $P(n,s)$ has unit row sums (and non-negative elements, indeed). Such a matrix is called a *stochastic matrix*. Homogeneity is an important property of Markov chains, we shall use it in Chapter 3 in the convergence study of Simulated Annealing.

Definition 1.7.6 (Homogeneous Chains) *If the conditional probabilities $P_{ij}(n,n+1)$ do not depend on the time parameter n, then the chain is called homogeneous. We shall denote the transition probabilities $P_{ij}(n,n+1)$ by P_{ij}.*

Assuming an homogeneous chain, let us examine the transition probability from a_j to a_k in exactly l steps. This can occur via various paths $a_j \to a_{j_1} \to \cdots \to a_{j_{l-1}} \to a_k$. The probability of such a path is $P_{jj_1}P_{j_1j_2}\cdots P_{j_{l-1}k}$. The sum P^l_{jk} of these particular probabilities gives the probability of finding the system in state a_k after l transitions, given the initial state a_j. Obviously, $P^1_{jk} = P_{jk}$ and

$$P^{n+1}_{jk} = \sum_i P_{ji}P^n_{ik}, \tag{1.95}$$

or more generally

$$P^{n+m}_{jk} = \sum_i P^m_{ji}P^n_{ik}. \tag{1.96}$$

Introducing a matrix-notation, the above equations can be rewritten as $P^{n+1} = PP^n$ and $P^{n+m} = P^nP^m$. The unconditional probability of having state a_k after n transitions is given by

$$p^n_k = \sum_j p_j P^n_{jk} \tag{1.97}$$

A state a_k can be *reached* from a_j if there exists some n such that $P^n_{jk} > 0$.

Definition 1.7.7 (Stationary Distribution) *If the unconditional probabilities p^n_k are independent of n, that is if*

$$\forall n:\ \ p^n_k = p_k \tag{1.98}$$

then the probability distribution p_k is called stationary.

Definition 1.7.8 (Irreducibility) *A Markov chain is irreducible if for all pairs of states (a_j, a_k) there is a positive probability of reaching a_k from a_j in a finite number of transitions:*

$$\forall a_j, a_k\ \exists n: \quad P^n_{jk} > 0. \tag{1.99}$$

Definition 1.7.9 (Aperiodicity) *A Markov chain is aperiodic if for all states a_i, the greatest common divisor of all integers $n \geq 1$, such that $P^n_{ii} > 0$ is equal to 1.*

Now, let us discuss ergodic properties of inhomogeneous Markov chains.

Definition 1.7.10 (Weak Ergodicity) *An inhomogeneous Markov chain is weakly ergodic if for all $m \geq 1$:*

$$\lim_{n\to\infty} (P_{ik}(m,n) - P_{jk}(m,n)) = 0 \tag{1.100}$$

Definition 1.7.11 (Strong Ergodicity) *An inhomogeneous Markov chain is strongly ergodic if there exists a vector π, satisfying:*

$$\sum_i \pi_i = 1, \quad \forall i : \pi_i > 0, \tag{1.101}$$

such that for all $m \geq 1$:

$$\lim_{n\to\infty} P_{ij}(m,n) = \pi_j. \tag{1.102}$$

In other words, *ergodicity* means that the transition probabilities converge to a distribution π. *Weak ergodicity* states only the convergence (remark that Equation (1.100) is nothing else but the well known *Cauchy-criterion*) and *strong ergodicity* states that they converge towards π. For a stationary distribution, π must coincide with the stationary distribution. In statistical physics, this convergence may be interpreted as a tendency towards a state of equilibrium. However, such a state of equilibrium does not always exist (for example in the case of a random walk presented in Example 1.7.1). Ergodicity is used in Chapter 3 for the convergence study of Simulated Annealing.

1.8 Markov Random Fields

In the 20's, mostly inspired by the Ising model, a new type of stochastic process appeared in the theory of probability called *Markov Random Field* (MRF). MRF's become rapidly a broadly used tool in a variety of problems not only in statistical mechanics. Its use in image processing became popular with the famous paper of S. Geman and D. Geman [46] in 1984 but its first use in the domain dates in the early 70's [14]. Here, we briefly give an introduction to the theory of MRF's [81, 105, 94, 36, 57].

1.8.1 The Ising Model

Following Ising, we consider a sequence, $0, 1, 2, \ldots, n$ on the line. At each point, there is a small spin which is either *up* or *down* at any given moment (see Figure 1.6).

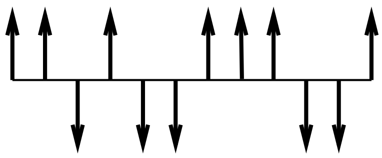

Figure 1.6: *One dimensional Ising model.*

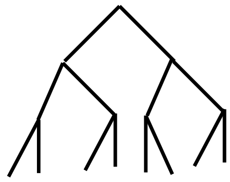

Figure 1.7: *Cayley tree model.*

Now, we define a probability measure on the set Ω of all possible configurations $\omega = (\omega_0, \omega_1, \dots, \omega_n)$. In this context, each spin is a function

$$\delta_i(\omega) = \begin{cases} 1 & \text{if } \omega_i \text{ is } \textit{up} \\ -1 & \text{if } \omega_i \text{ is } \textit{down} \end{cases} \tag{1.103}$$

An *energy* $U(\omega)$ is assigned to each configuration:

$$U(\omega) = -J \sum_{i,j} \delta_i(\omega)\delta_j(\omega) - mH \sum_i \delta_i(\omega) \tag{1.104}$$

In the first sum, Ising made a simplifying assumption that only interactions of points with one unit apart need to be taken into account. This term represents the energy caused by the spin-interactions. The constant J is a property of the material. If $J > 0$, the interactions tend to keep neighboring spins in the same directions (*attractive case*). If $J < 0$, neighboring spins with opposite orientation are favored (*repulsive case*). The second term represents the influence of an external magnetic field of intensity H and $m > 0$ is a property of the material. The probability measure on Ω is then given by

$$P(\omega) = \frac{\exp\left(-\frac{1}{kT}U(\omega)\right)}{Z}, \tag{1.105}$$

where T is the temperature and k is a universal constant. The normalizing constant (or *partition function*) Z is defined by

$$Z = \sum_{\omega \in \Omega} \exp\left(-\frac{1}{kT}U(\omega)\right). \tag{1.106}$$

The probability measure defined in Equation (1.105) is called a *Gibbs distribution*. One could extend the model to two dimensions in a natural way. The spins are arranged on a lattice, they are represented by two coordinates and a point have 4 neighbors unless it is on the boundary. In the two-dimensional case, the limiting measure P is unstable, there is a *phase transition*. As it is pointed out in [81], considering the *attractive* case and an external field h, the measure P_h converges to P^- if h goes to zero through negative values but it converges to $P^+ \neq P^-$ if h goes to zero through positive values. It has been shown, that there exists a *critical temperature* T_C and below this temperature phase transition always occurs. The temperature depends on the vertical (J_1) and horizontal (J_2) interaction parameters.

As a special example, we mention the *Cayley tree model* [81], originally proposed by Bethe [9] as an approximation to the Ising model. In this case, the points sit on a tree (see Figure 1.7). The root is called the 0^{th} level. From the root, we have q branches ($q = 2$ in Figure 1.7). The $q = 1$ case simply gives a one dimensional Markov chain. A configuration on a tree of n levels is an assignment of a label *up* or *down* to each point. We can define a similar energy function as for the Ising model.

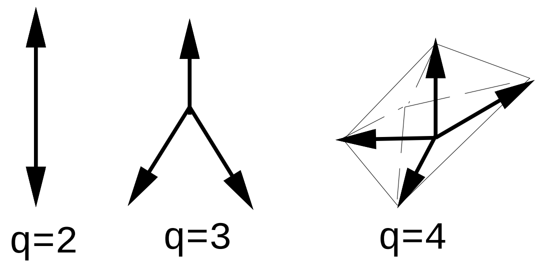

Figure 1.8: *The Potts model.*

Another extension of the Ising model to more than two states per points is the Potts model [111, 106]. The problem is to regard the Ising model as a system of interacting spins that can be either parallel or antiparallel. More generally, we consider a system of spins, each spin pointing one of the q equally spaced directions. These vectors are the linear combinations of q unit vectors pointing in the q symmetric directions of a hypertetrahedron in $q-1$ dimensions. For $q = 2, 3, 4$, examples are shown in Figure 1.8. The energy function of the Potts model can be written as

$$U(\omega) = \sum_{i,j} J(\Theta_{ij}), \tag{1.107}$$

where $J(\Theta)$ is 2π periodic and Θ_{ij} is the angle between two neighboring spins in i and j. The $q = 2$ case is equivalent to the Ising model.

1.8.2 Gibbs Distribution and MRF's

The most natural way to define MRF's [3, 105, 46] related to image models is to define them on a lattice. However, here we will define MRF's more generally on graphs. It will be useful in Chapter 2 for the study of hierarchical models. Let $\mathcal{G} = (\mathcal{S}, \mathcal{E})$ be a graph where $\mathcal{S} = \{s_1, s_2, \ldots, s_N\}$ is a set of vertices (or sites) and $\mathcal{E}$ is the set of edges.

Definition 1.8.1 (Neighbors) *Two points s_i and s_j are neighbors if there is an edge $e_{ij} \in \mathcal{E}$ connecting them. The set of points which are neighbors of a site s (ie. the neighborhood of s) is denoted by $\mathcal{V}_s$.*

Definition 1.8.2 (Neighborhood system) $\mathcal{V} = \{\mathcal{V}_s \mid s \in \mathcal{S}\}$ *is a neighborhood system for $\mathcal{G}$ if*

(i) $s \notin \mathcal{V}_s$

(ii) $s \in \mathcal{V}_r \Leftrightarrow r \in \mathcal{V}_s$

To each site of the graph, we assign a label λ from a finite set of labels Λ. Such an assignment is called a configuration ω having some probability $P(\omega)$. The restriction to a subset $T \subset \mathcal{S}$ is denoted by ω_T and $\omega_s \in \Lambda$ denotes the label given to the site s. In the following, we are interested in the probability measures assigned to the set Ω of all possible configurations. First, let us define the *local characteristics* as the conditional probabilities $P(\omega_s \mid \omega_r, r \neq s)$.

Definition 1.8.3 (Markov Random Field) *$\mathcal{X}$ is a Markov Random Field (MRF) with respect to $\mathcal{V}$ if*

(i) for all $\omega \in \Omega$: $P(\mathcal{X} = \omega) > 0$,

(ii) for every $s \in \mathcal{S}$ and $\omega \in \Omega$:
$P(X_s = \omega_s \mid X_r = \omega_r, r \neq s) = P(X_s = \omega_s \mid X_r = \omega_r, r \in \mathcal{V}_s)$.

To continue our discussion about probability measures on Ω, the notion of *cliques* will be very useful.

Definition 1.8.4 (Clique) *A subset $C \subseteq \mathcal{S}$ is a clique if every pair of distinct sites in C are neighbors. $\mathcal{C}$ denotes the set of cliques and* $\deg(\mathcal{C}) = \max_{C \in \mathcal{C}} \mid C \mid$.

Using the above definition, we can define a *Gibbs measure* on Ω. Let V be a *potential* which assign a number $V_T(\omega)$ to each subconfiguration ω_T. V defines an *energy* $U(\omega)$ on Ω by

$$U(\omega) = -\sum_T V_T(\omega). \tag{1.108}$$

Definition 1.8.5 (Gibbs distribution) *A Gibbs distribution is a probability measure π on Ω with the following representation:*

$$\pi(\omega) = \frac{1}{Z} \exp\left(-U(\omega)\right), \tag{1.109}$$

where Z is the normalizing constant or partition function:

$$Z = \sum_{\omega} \exp\left(-U(\omega)\right),$$

If $V_T(\omega) = 0$ whenever T is not a clique then V is called a *nearest neighbor Gibbs potential.* In the following, we will focus on such potentials. The next famous theorem establish the equivalence between Gibbs measures and MRF's [14, 94].

Theorem 1.8.1 (Hammersley-Clifford) *$\mathcal{X}$ is a MRF with respect to the neighborhood system $\mathcal{V}$ if and only if $\pi(\omega) = P(\mathcal{X} = \omega)$ is a Gibbs distribution with a nearest neighbor Gibbs potential V, that is*

$$\pi(\omega) = \frac{1}{Z} \exp\left(-\sum_{C \in \mathcal{C}} V_C(\omega)\right) \tag{1.110}$$

The main benefit of this equivalence is that it provides us a simple way to specify MRF's, namely specifying potentials instead of local characteristics (see Definition 1.8.3), which is usually very difficult.

1.8.3 Spatial Lattice Schemes

In this section, we deal with a particular subclass of MRF's which are the most commonly used schemes in image processing. In this case, we consider $\mathcal{S}$ as a lattice $\mathcal{L}$ so that $\forall s \in \mathcal{S} : s = (i, j)$ and define the so-called n^{th} *order homogeneous neighborhood systems* as

$$\mathcal{V}^n = \{\mathcal{V}^n_{(i,j)} : (i,j) \in \mathcal{L}\}, \tag{1.111}$$

$$\mathcal{V}^n_{(i,j)} = \{(k,l) \in \mathcal{L} : (k-i)^2 + (l-j)^2 \leq n\}. \tag{1.112}$$

Obviously, sites near the boundary have fewer neighbors than interior ones (free boundary condition). Furthermore, $\mathcal{V}^0 \equiv \mathcal{S}$ and for all $n \geq 0 : \mathcal{V}^n \subset \mathcal{V}^{n+1}$. Figure 1.9 shows a first-order neighborhood corresponding to $n = 1$. The cliques are $\{(i,j)\}, \{(i,j),(i,j+1)\}, \{(i,j),(i+1,j)\}$. In practice, more than two order systems are rarely used since the energy function would be too complicated and will require a lot of computation.

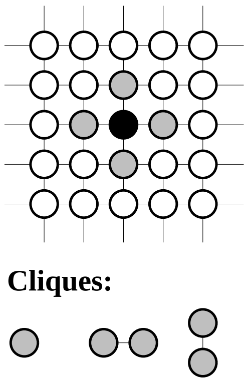

Figure 1.9: *First order neighborhood system.*

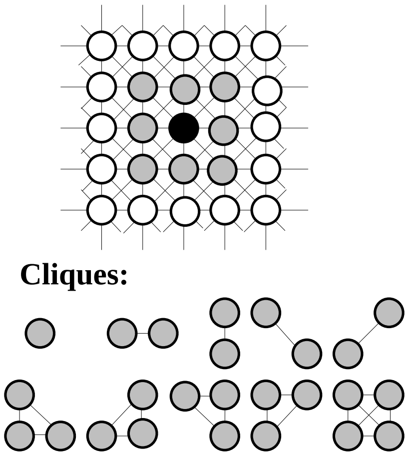

Figure 1.10: *Second order neighborhood system.*

In This Chapter:

2.

Image Models

Early vision refers to a variety of digital image processing tasks dealing directly with massive amounts of pixel data. The goal of such a process is to transform the digitized image data into more meaningful tokens (edges, texture features, regions, etc...).

The variety of early vision tasks has resulted in a variety of algorithms, sometimes dedicated to a single application and often tuned to a particular environment in which they are implemented. A general framework in low level vision is image labeling where we want to associate to each pixel a label from a finite set. The meaning of this label depends on the problem that we are trying to solve. For image restoration, it means a grey-level; for edge detection, it means the presence or the direction of an edge; for image segmentation, it means a class (or region); etc... The problem here is how to choose a label for a pixel. There may be various responses. Our approach consists of building probabilistic image models and simply selecting the most likely labeling. To do this, we need to define some probability measure on the set of all possible labelings. In real scenes, neighboring pixels usually have similar intensities. In a probabilistic framework, such regularities are well expressed mathematically by Markov Random Fields (MRF). Another reason for dealing with MRF models is of course the *Hammersley-Clifford theorem* which allows us to define MRF's through clique-potentials. In the labeling problem, this leads us to the Bayesian formulation in which a prior distribution is needed on the labels. In general, we try to find the Maximum A Posteriori (MAP) estimate of the label field.

Unfortunately, finding such an estimate is a heavy computational problem. There are many heuristics to make the minimization easier such as multi-scale approaches. In the next sections, after introducing the classical monogrid model, we present some multiscale models proposed by a variety of authors. Finally, we propose a new type of MRF model called hierarchical MRF model. This model allows to work with cliques with far apart sites for a reasonable price.

2.1 Related Non-Markovian Approaches

2.1.1 Relaxation Labeling

As we said in the introduction, many problems in computer vision can be formulated in terms of the assignment of labels to objects. Processes are needed which select the best label among several possible choices. A well known, non-Markovian approach of the problem is the so-called Relaxation Labeling (RL) process [67, 38]. Although, there are similarities with the Markovian approach, a fundamental difference is that RL is a non-stochastic process.

Let us first briefly discuss RL. We are given a set of objects, a set of labels for each object (or possibly a common set for all objects), a neighbor relation over the objects, and a *constraint relation* over labels at pairs (or n-tuples) of neighboring objects. A solution to the problem is a labeling which is consistent with respect to the *constraint relation*. Let us denote the objects (or nodes if we consider the objects siting on a graph) by $i = 0, \ldots n$, and the corresponding set of labels by Λ_i. The constraints Λ_{ij} is the set of all pairs of labels in $\Lambda_i \times \Lambda_j$ which are compatible at the neighboring objects i and j. Assuming discrete relaxation, a label is discarded at node i if there is no compatible label at the neighboring nodes and we keep it if there is at least one compatible label. In the continuous case, we use weights for label assignments. Denoting the weight with which λ is assigned to node i by $p_i(\lambda)$, we require that

$$\forall i, \lambda\colon\ \ 0 \leq p_i(\lambda) \leq 1 \tag{2.1}$$

and

$$\sum_\lambda p_i(\lambda) = 1. \tag{2.2}$$

Similarly, the constraints are generalized to real-value compatibility functions $r_{ij}(\lambda, \lambda')$ signifying relative support of label λ at i given the label at j is λ'. The label weights can be regarded as probabilities and describe the RL process in Bayesian terms. However such tentations have been limited and unsuccessful within the RL framework. The goal of all RL processes is to find a consistent labeling. In [67], consistency is defined by a system of inequalities permitting the constraints to be ordered with respect to their relative importance.

The similarities of the MRF approach with RL are obvious: We have a graph (which is usually a lattice) and we have constraints on neighboring objects (pixels) of the graph. However, in a Markovian model, the constraints are expressed by local conditional probabilities through clique-potentials. The *consistency* of a labeling is also expressed in a probabilistic way: we are looking for a Bayesian estimate of the label field. In summary, we can say that while there are shared goals and features (graph structure, neighborhoods, locality, etc...), RL and MRF models are still quite distinct approaches.

2.1.2 Weak Membrane Model

The *weak membrane model* have been introduced in image reconstruction by A. Blake and A. Zisserman [15]. The problem is to reconstruct surfaces which are *continuous almost everywhere* or, in other words, continuous in patches. To reach a satisfactory formalization of this principle, they have developed a membrane model: Imagine an elastic membrane which we are trying to fit to a surface The edges will appear as tears in the membrane. Depending on how elastic is the membrane, there may be more or less edges. The membrane is described by an energy function (the elastic energy of the membrane) which has to be minimized in order to find an equilibrium state. The energy has three components:

D: A measure of faithfulness to the data:

$$D = \int (u - d)^2 dA \tag{2.3}$$

where $u(x, y)$ represents the membrane and $d(x, y)$ represents the data.

S: A measure of how the function $u(x, y)$ is deformed:

$$S = \lambda^2 \int (\nabla u)^2 dA. \tag{2.4}$$

λ^2 is a measure of elasticity of the membrane.

P: The sum of penalties α levied for each break in the membrane:

$$P = \alpha Z, \tag{2.5}$$

where Z is a measure of the set of contours along which $u(x, y)$ is discontinuous (see [15] for more details).

The elastic energy of the membrane is then given by

$$E = D + S + P = \int (u - d)^2 dA + \lambda^2 \int (\nabla u)^2 dA + \alpha Z. \tag{2.6}$$

There is a strong relation between the *weak membrane model* and MRF models: An elastic system can also be considered from a probabilistic view-point. The link between the elastic energy E and probability P is

$$P \propto \exp\left(\frac{-E}{T}\right), \tag{2.7}$$

that is the Gibbs distribution. However, the *weak membrane model* operates with mechanical analogies, representing *a priori* knowledge from a mechanical point of view while MRF modelization is purely probabilistic[1].

2.2 A General Markovian Image Model

MRF models in computer vision has become popular with the famous paper of S. Geman and D. Geman on image restoration [46]. The field has grown up rapidly in recent years addressing a variety of low-level[2] image tasks [2]:

Compression: Find a new image as close as possible to the original one but described at a much smaller cost (MRF's are used in other compression problems too).

Restoration: Observing a degraded image, one wants to approximately recover the original one.

Edge Detection: Find smooth boundaries separating image regions.

Segmentation: Partition the image into homogeneous regions where homogeneity is measured in terms of grey-levels or texture characteristics.

Motion Detection: In a sequence of images, try to find a field of velocities linking one image to the next one.

We now turn to the mathematical formulation of a MRF image model. Let $\mathcal{R} = \{r_1, r_2, \ldots, r_M\}$ be a set of sites and $\mathcal{F} = \{F_r : r \in \mathcal{R}\}$ a set of image data (or observations) on these sites. The set of all possible observations $f = (f_{r_1}, f_{r_2}, \ldots, f_{r_M})$ is denoted by Φ. Furthermore, we are given another set of sites $\mathcal{S} = \{s_1, s_2, \ldots, s_N\}$, each of these sites may take a label from $\Lambda = \{0, 1, \ldots, L-1\}$. The configuration space Ω is the set of all global discrete labeling $\omega = (\omega_{s_1}, \ldots, \omega_{s_N}), \omega_s \in \Lambda$. The two set of

[1] We notice that the weak membrane model has also been used in a Markovian context but originally, as proposed by Blake and Zisserman [15], it was a non-Markovian model

[2] *Low-level* is a traditional terminology for preliminary tasks to image understanding.

sites $\mathcal{R}$ and $\mathcal{S}$ are not necessarily disjunct, they may have common parts (for example Geman's image restoration model involving a line process [46]) or refer to a common set of sites. Our goal is to modelize the labels and observations with a joint random field $(\mathcal{X}, \mathcal{F}) \in \Omega \times \Phi$. The field $\mathcal{X} = \{X_s\}_{s\in\mathcal{S}}$ is called the *label field* and $\mathcal{F} = \{F_r\}_{r\in\mathcal{R}}$ is called the *observation field.*

2.2.1 Bayesian estimation

First, we construct a Bayesian estimator for the *label field.* Using the results from Section 1.3, we can define both the joint and conditional probabilities in terms of the a priori and a posteriori distributions:

$$P_{\mathcal{X},\mathcal{F}}(\omega, f) = P_{\mathcal{F}|\mathcal{X}}(f \mid \omega) P_{\mathcal{X}}(\omega) \tag{2.8}$$

$$P_{\mathcal{X}|\mathcal{F}}(\omega \mid f) = \frac{P_{\mathcal{X},\mathcal{F}}(\omega, f)}{P_{\mathcal{F}}(f)} = \frac{P_{\mathcal{F}|\mathcal{X}}(f \mid \omega) P_{\mathcal{X}}(\omega)}{P_{\mathcal{F}}(f)} \tag{2.9}$$

Since the realization of the observation field is known, $P(f)$ is constant and we can write:

$$P_{\mathcal{X}|\mathcal{F}}(\omega \mid f) \propto P_{\mathcal{F}|\mathcal{X}}(f \mid \omega) P_{\mathcal{X}}(\omega) \tag{2.10}$$

The estimator is the following decision function δ (see Section 1.6):

$$\delta \;:\; \Phi \longrightarrow \Omega \tag{2.11}$$

$$f \mapsto \delta(f) = \hat{\omega} \tag{2.12}$$

and the corresponding *Bayes risk* is given by

$$r(P_X, \delta) = E\{R(\omega, \delta(f))\} \tag{2.13}$$

where $R(\omega, \delta(f))$ is a cost function. According to the Definition 1.6.7, our estimator must have the minimum Bayes risk:

$$\hat{\omega} = \arg\min_{\omega'\in\Omega} \int_{\omega\in\Omega} R(\omega, \omega') P_{\mathcal{X}|\mathcal{F}}(\omega \mid f) d\omega \tag{2.14}$$

We explain hereafter the three best known Bayesian estimators [89].

2.2.1.1 Maximum A Posteriori (MAP)

The MAP estimator is the most frequently used estimator in image processing. Its cost function is defined by

$$R(\omega, \omega') = 1 - \Delta_{\omega'}(\omega), \tag{2.15}$$

where $\Delta_{\omega'}(\omega)$ is the Dirac mass in ω'. Clearly, this function has the same cost for all configurations different from ω'. From Equation (2.14) and Equation (2.15), the MAP estimator of the label field is given by

$$\hat{\omega}^{MAP} = \arg\max_{\omega \in \Omega} P_{\mathcal{X}|\mathcal{F}}(\omega|f). \tag{2.16}$$

This estimator gives for a given observation f, the modes of the posterior distribution, that is the most likely labelings given the observation f. Equation (2.16) is a combinatorial optimization problem which requires special algorithms such as Simulated Annealing (see Chapter 3).

2.2.1.2 Marginal A Posteriori Modes (MPM)

We define the cost function of the MPM estimator as

$$R(\omega, \omega') = \sum_{s \in \mathcal{S}} (1 - \Delta_{\omega'_s}(\omega_s)). \tag{2.17}$$

Remark that the above function is related to the number of sites $s \in \mathcal{S}$ such that $\omega_s \neq \omega'_s$. The solution of Equation (2.14) is given by

$$\forall s \in \mathcal{S} : \hat{\omega}_s^{MPM} = \arg\max_{\omega_s \in \Lambda} P_{X_s|\mathcal{F}}(\omega_s \mid f), \tag{2.18}$$

which gives the configuration which maximizes at each site the a posteriori marginal $P_{X_s|\mathcal{F}}(. \mid f)$.

2.2.1.3 Mean Field (MF)

Here, we have the following cost function:

$$R(\omega, \omega') = \sum_{s \in \mathcal{S}} (\omega_s - \omega'_s)^2. \tag{2.19}$$

From Equation (2.14) and Equation (2.19), we have

$$\forall s \in \mathcal{S} : \hat{\omega}_s^{MF} = \int_{\omega \in \Omega} \omega_s P_{\mathcal{X}|\mathcal{F}}(\omega \mid f) d\omega, \tag{2.20}$$

which is nothing else but the conditional expected value of $\mathcal{X}$ given $\mathcal{F} = f$ that is the *mean field* of $\mathcal{X}$.

2.2.2 Defining a Priori and a Posteriori Distributions

In the Bayesian framework, our knowledge about the "world" is represented by a priori probabilities. However, in practice, it is extremely difficult to define such probabilities globally, even if we focus on a specific area of image processing. But there are some well defined properties if we are considering images *locally*: Usually, neighboring pixels have similar intensities, edges are smooth and often straight and textures have also well defined local properties. It is then a better idea to represent our knowledge in terms of some local random variables. This kind of knowledge is best described by means of MRF's.

2.2.2.1 Prior Distribution

Let us suppose that $\mathcal{X}$ is a MRF with some neighborhood system $\mathcal{V}' = \{\mathcal{V}'_s : s \in \mathcal{S}\}$ and distribution

$$
\begin{aligned}
P(\mathcal{X} = \omega) &= \frac{1}{Z} \exp\left(-U'(\omega)\right), &(2.21)\\
U'(\omega) &= \sum_{C \in \mathcal{C}'} V'_C(\omega) &(2.22)
\end{aligned}
$$

where $U'(\omega)$ is the energy function (see Section 1.8). The above equations give another good reason using MRF priors, namely their Gibbs representation through *clique-potentials*, which are more convenient than working directly with probabilities.

2.2.2.2 Degraded Image Model and Posterior Distribution

The observations are related to the label process through a *degradation model* which models the relation between the *label field* $\mathcal{X}$ and the *observation process* $\mathcal{F}$. In image restoration [46] for example, what we observe is a blurred noisy image and we want to restore the original one. So, the label process represents grey-levels here. We are now considering a similar model but in a more general manner. Most of the problems result in the following function [97]:

$$\mathcal{F} = \Psi(H(\mathcal{X}), N), \tag{2.23}$$

or at the pixel level:

$$\forall r \in \mathcal{R} : F_r = \Psi(H_r(X_{\psi(r)}), N_r) \tag{2.24}$$

where $\Psi(a, b)$ is an invertible function in a. H_r is a local function defined on a small part $\psi(r)$ of $\mathcal{S}$ such that $\psi(r) \in \mathcal{S}, \mid \psi(r) \mid \ll \mid \mathcal{S} \mid$ and $\psi^{-1}(s) = \{r \in \mathcal{R} \mid s \in \psi(r)\}$.

N is a random component (usually a Gaussian white noise but in tomography N_r are Poisson variables whose means are related to $\mathcal{X}$). In [46], for instance, H is a *blurring matrix* and N is an additive Gaussian noise. If we assume that the distribution of N is given by

$$P_N(.) = \prod_{r \in \mathcal{R}} P_{N_r}(.) \tag{2.25}$$

then we obtain

$$P_{\mathcal{F}|\mathcal{X}}(f \mid \omega) = \prod_{r \in \mathcal{R}} P_{N_r}(\Psi^{-1}(H_r(\omega_{\psi(r)}), f_r)). \tag{2.26}$$

The conditional distribution of the observation field $\mathcal{F}$ given $\mathcal{X}$ can be written as

$$P_{\mathcal{F}|\mathcal{X}}(f \mid \omega) = \exp\left(\sum_{r \in \mathcal{R}} -\ln(P_{N_r}(\Psi^{-1}(H_r(\omega_{\psi(r)}), f_r)))\right), \tag{2.27}$$

assuming that $P_{N_r}(.) > 0$ at each site r in $\mathcal{R}$. Combining the above equation with Equation (2.10) and Equation (2.21), the posterior distribution is of the following form:

$$P_{\mathcal{X}|\mathcal{F}}(\omega \mid f) \propto \frac{1}{Z} \exp\left(\sum_{r \in \mathcal{R}} -\ln(P_{N_r}(\Psi^{-1}(H_r(\omega_{\psi(r)}), f_r))) + \sum_{C \in \mathcal{C}'} V'_C(\omega)\right) \tag{2.28}$$

Notice that the posterior distribution is also a Gibbs distribution with the smallest neighborhood system $\mathcal{V}$ containing all the cliques in $\mathcal{C}'$ and the sets $\{\psi(r), r \in \mathcal{R}\}$:

$$\forall s \in \mathcal{S} : \mathcal{V}_s = \left(\bigcup_{r \in \psi^{-1}(s)} \psi(r) \setminus \{s\}\right) \cup \mathcal{V}'_s \tag{2.29}$$

Let us denote the corresponding energy function by $U(\omega, f)$:

$$\begin{aligned} U(\omega, f) &= \sum_{r \in \mathcal{R}} -\ln(P_{N_r}(\Psi^{-1}(H_r(\omega_{\psi(r)}), f_r))) + \sum_{C \in \mathcal{C}'} V'_C(\omega) \\ &= \sum_{r \in \mathcal{R}} V_r(\omega_{\psi(r)}), f_r) + \sum_{C \in \mathcal{C}'} V'_C(\omega) \end{aligned} \tag{2.30}$$

In the following, we will be more specific about $V_r(\omega_{\psi(r)}), f_r)$ and suppose that it is of the form [97]:

$$V_r(\omega_{\psi(r)}), f_r) = V_r(\omega_{\psi(r)}) + \sum_{s \in \psi(r)} V_{s,r}(\omega_s, f_r). \tag{2.31}$$

This restriction is less severe than it might be expected. As we will see, most of the nowadays used models have this kind of energy function. The above equation can be rewritten as

$$\begin{aligned}\sum_{r\in\mathcal{R}} V_r(\omega_{\psi(r)}), f_r) &= \sum_{r\in\mathcal{R}} V_r(\omega_{\psi(r)}) + \sum_{r\in\mathcal{R}} \sum_{s\in\psi(r)} V_{s,r}(\omega_s, f_r) &(2.32)\\ &= \sum_{r\in\mathcal{R}} V_r(\omega_{\psi(r)}) + \sum_{s\in\mathcal{S}} \underbrace{\sum_{r\in\psi^{-1}(r)} V_{s,r}(\omega_s, f_r)}_{V_s(\omega_s, f_{\psi^{-1}(s)})} &(2.33)\\ & &(2.34)\end{aligned}$$

Finally, we have the following energy function associated with the posterior distribution of the label field $\mathcal{X}$:

$$\begin{aligned}U(\omega, f) &= \sum_{s\in\mathcal{S}} V_s(\omega_s, f_{\psi^{-1}(s)}) + \sum_{C\in\mathcal{C}} V_C(\omega) &(2.35)\\ &= U_1(\omega_s, f_{\psi^{-1}(s)}) + U_2(\omega). &(2.36)\end{aligned}$$

where the clique-potentials $V_C(\omega)$ are defined as

$$V_C(\omega) = \begin{cases} V'_C(\omega) & \text{if } C \in \mathcal{C}' \text{ and } C \notin \{\psi(r), r \in \mathcal{R}\} \\ V_r(\omega_{\psi(r)}) & \text{if } C = \psi(r) \text{ and } \psi(r) \notin \mathcal{C}' \\ V'_C(\omega) + V_r(\omega_{\psi(r)}) & \text{if } C = \psi(r) \text{ and } \psi(r) \in \mathcal{C}' \end{cases} \quad (2.37)$$

If we assume that the observed image $\mathcal{F}$ is affected at site s only by the pixel s itself then Equation (2.35) can be further simplified: $\psi(r)$ reduces to s and the neighborhood system of the posterior distribution is equivalent to the one of the prior distribution.

2.2.3 Some Examples of Markov Models

Herein, we explain some models applied to a variety of image processing tasks. Most of them uses the general model discussed in the previous section (see Equation (2.35)). Let us begin this discussion with the restoration model proposed by D. Geman and S. Geman in [46].

2.2.3.1 Image Restoration

We observe a blurred noisy image $\mathcal{F}$ and we want to restore the original one. The components of the degraded model in Equation (2.23) have the following meanings: N is supposed to be a white Gaussian noise with mean μ and variance σ^2. H is a shift-invariant blurring matrix. First of all, we define the lattices on which the label process and the observation process are defined. The *observation process* simply consists of the grey-level at each pixel of the given image. Thus $\mathcal{R}$ is a lattice, each site corresponds to a pixel. The *label process* is more sophisticated involving both *pixel sites* and *line sites*. $\mathcal{X}$ is then a "mixed" process having two subprocesses: a *pixel process* and a *line process*. The lattice $\mathcal{S}$ contains $\mathcal{R}$ (*pixel sites*) and another lattice with sites siting between each vertical and horizontal pair of pixels representing a possible location of edge elements (*line sites*, see Figure 2.1).

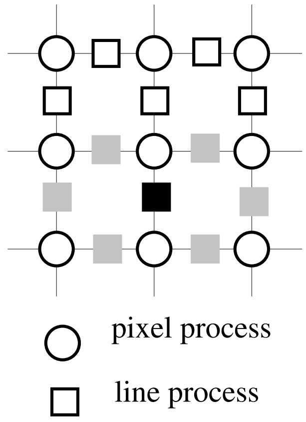

Figure 2.1: *Geman's image restoration model*

We now turn to the posterior distribution and its energy function. Let us denote the *line process* by $\mathcal{X}^l$ and the *pixel process* by $\mathcal{X}^p$. We assume that $\mathcal{X}^p$ is a MRF over a homogeneous neighborhood system (see Section 1.8.3) $\mathcal{V}$ on $\mathcal{R}$ and $\mathcal{X}^l$ is also a MRF over a neighborhood system shown on Figure 2.1. (the neighbors of the black site are the grey sites). $\mathcal{X}$ has a prior distribution of

$$P(\mathcal{X}^p = \omega^p, \mathcal{X}^l = \omega^l)$$

$$= \frac{1}{Z} \exp(-U'(\omega^p, \omega^l)) = \frac{1}{Z} \exp\left(-\sum_{C \in \mathcal{C}} V_C(\omega)\right), \qquad (2.38)$$

where $\omega = (\omega^p, \omega^l)$. ω^p takes values among the available grey-levels and ω^l among the line states. If we choose $\mathcal{V}$ such that it is large enough to encompass the dependencies caused by the blurring H then the posterior distribution also defines a MRF with energy function

$$U(\omega^p, \omega^l) = U'(\omega^p, \omega^l) + \frac{\|\vec{\mu} - \Psi^{-1}(H(\omega^p), f)\|^2}{2\sigma^2} \qquad (2.39)$$

The optimal labeling $\hat{\omega}$ is found by the MAP estimate minimizing the above energy function. The restored image is then given by the *pixel process* $\hat{\omega}^p$.

2.2.3.2 Texture Segmentation

The observations consist of a set of various texture features describing spatial statistics of the image. These features are computed on local windows around each pixel including mean, variance, correlation, entropy, contrast, homogeneity, etc [61, 48, 47, 33, 28, 29]...Here, both the *observation process* and the *label process* are defined on the same lattice $\mathcal{S}$ with sites corresponding to image pixels. The terms U_1 and U_2 from Equation (2.36) are defined in the following way: The prior energy U_2 usually favors spatially homogeneous regions assigning lower potentials to homogeneous cliques. The term U_1 does not have such a "standard" definition. It has various form in the literature. In [61], it measures the distance, at a given point s, between the distribution of the texture features in a small block B_s centered at s and the one in the whole (candidate) region R_s to which we want to assign s. This technique, as claimed in [61], permits to automatically determine the number of regions. The energy function is defined as

$$\begin{aligned} U_1(\omega, \vec{f}) &= \sum_{s \in \mathcal{S}} V_s(B_s, R_s) && (2.40) \\ V_s(B_s, R_s) &= \sum_{i=1}^{m} (2\Delta(d(\vec{f}^{\,i}_{R_s}, \vec{f}^{\,i}_{B_s}) > c^i) - 1) && (2.41) \end{aligned}$$

where m is the number of considered features. $\vec{f}_{B_s}$ and $\vec{f}_{R_s}$ denote the set of feature vectors on block B_s and on the region R_s respectively. $d(a, b)$ stands for the *Kolmogorov-Smirnov distance* and c^i is a threshold given by statistical tables associated to the Kolmogorov limit distribution (see [61]). The function Δ returns 1 if its argument is true, 0 otherwise.

2.2.3.3 Edge Detection

MRF models for edge detection are often *compound Gauss-Markov random fields* (CGMRF) [69, 115]. The local characteristics of a CGMRF is given by

$$P(f_s \mid f_r, r \in \mathcal{S}) = \frac{1}{\sqrt{2\pi}\varsigma} \exp\left(\frac{1}{2\varsigma^2} \left(f_s - \mu_m - \sum_{r \in \mathcal{V}_s} \vartheta_r (f_r - \mu_m) \right)^2 \right), \qquad (2.42)$$

where ς is the deviation, μ_m is the mean and ϑ_r is the model parameter. The supporting graph is similar to the one reported in [46] (cf. Figure 2.1). The observations $\mathcal{F}$ are considered to be corrupted by an additive Gaussian noise with zero mean and variance σ^2. The label field is again a mixed process containing both a *pixel process* $\mathcal{X}^p$ and a *line process*. Assuming a first order neighborhood system (cf. Figure 1.9) and denoting the horizontal and vertical line process by $\mathcal{X}^h$ and $\mathcal{X}^v$ respectively, a possible form of the energy function is given by [115]:

$$\begin{aligned} U(\omega, f) &= \frac{1}{2\sigma^2} \sum_{s=(i,j) \in \mathcal{S}} \Big((f_{i,j} - \omega^p_{i,j})^2 + \beta^2 (1 - 2(\vartheta_h + \vartheta_v)) f^2_{i,j} \\ &+ \vartheta_h (\beta^2 (f_{i,j} - f_{i-1,j})^2 (1 - \omega^h_{i,j}) + \alpha \omega^h_{i,j}) \\ &+ \vartheta_v (\beta^2 (f_{i,j} - f_{i,j+1})^2 (1 - \omega^v_{i,j}) + \alpha \omega^v_{i,j}) \Big) \qquad (2.43) \end{aligned}$$

with $1 - 2(\vartheta_h + \vartheta_v) > 0$. ϑ_v and ϑ_h are the model parameters for the vertical and horizontal cliques. β^2 corresponds to a regularization term reflecting the confidence in the data. In [115], $\beta^2 = \sigma^2/\varsigma^2$ expressing that when $\mathcal{F}$ is very noisy, we have no confidence in the data (β^2 is high). This model is related to the weak membrane model presented in [15]. The estimation of the *line process* is done by a Mean Field approach (see Section 2.2.1).

2.2.3.4 Motion Analysis

In [99], a MRF model for motion detection is presented. The *observation process* is defined both on the image-lattice $\mathcal{S}$ and on a time axis t. The detection of moving objects relies on the analysis of the variation of the intensity distribution in time. At each pixel, we have a two-element observation vector:

$$\vec{f}_s^1(t) = \mid y_s(t) - y_s(t - dt) \mid, \tag{2.44}$$

where $y_s(t)$ stands for the intensity value at pixel s at time t. $\vec{f}^2$ is a logical map of temporal changes between time t and $t - dt$. It equals to 1 if a temporal change of the intensity is valid at site s and 0 otherwise (for more details, see [64]). The *label process* is binary valued ($X_s(t) = 1$ if s is on a mask of a mobile object at time t). The energy function U consists of three terms. Two of them related to the observations and labels simultaneously (taking the role of U_1 in Equation (2.36)). One of them is used to reconstruct the mask of a mobile object at a given time:

$$U_1^2(\omega, \vec{f}_s^2) = \sum_{s \in \mathcal{S}} V_1(\omega_s(t), \vec{f}_s^2(t), \vec{f}_s^2(t + dt)), \tag{2.45}$$

the other expresses consistency between the current labeling and the intensity variation:

$$U_1^1(\omega, \vec{f}_s^1) = \sum_{s \in \mathcal{S}} \left(\frac{1}{2\sigma^2} (\vec{f}_s^1(t) - \mu\omega_s(t))^2 + (\vec{f}_s^1(t + dt) - \mu\omega_s)^2 \right) \tag{2.46}$$

where μ and σ are model parameters. The third term of the energy function U corresponds to U_2 with potentials favoring homogeneous masks.

2.3 An Image Segmentation Model

In this section, we discuss the image segmentation model used in our experiments. The model is very simple without any *line process* or texture. Our goal was not to establish a sophisticated, environment-specific model but to construct an easily applicable universal model for non-textured images and then study its multi-scale and hierarchical implementations.

Let us suppose that the observations consist of the grey-levels. A very general problem is to find the labeling $\hat{\omega}$ which maximizes the a posteriori probability $P(\omega \mid \mathcal{F})$. According to the results reported in Section 2.2.1, $\hat{\omega}$ is the MAP estimate of the *label field*. Bayes theorem tells us that

$$P(\omega \mid \mathcal{F}) = \frac{1}{P(\mathcal{F})} P(\mathcal{F} \mid \omega) P(\omega). \tag{2.47}$$

Actually $P(\mathcal{F})$ does not depend on the labeling ω and we make the assumption that

$$P(\mathcal{F} \mid \omega) = \prod_{s \in \mathcal{S}} P(f_s \mid \omega_s). \tag{2.48}$$

It is then easy to see that the global labeling, which we are trying to find, is given by:

$$\hat{\omega} = \arg\max_{\omega \in \Omega} \prod_{s \in \mathcal{S}} P(f_s \mid \omega_s) \prod_{C \in \mathcal{C}} \exp(-V_C(\omega_C)) \,. \tag{2.49}$$

It is obvious from this expression that the a posteriori probability also derives from a MRF. The energies of cliques of order 1 directly reflect the probabilistic modeling of labels without context, which would be used for labeling the pixels independently. Let us assume that $P(f_s \mid \omega_s)$ is Gaussian, the class $\lambda \in \Lambda = \{0, 1, \ldots, L-1\}$ is represented by its mean value μ_λ and its deviation σ_λ. We get the following energy function (using Equation (2.36)):

$$U_1(\omega, \mathcal{F}) = \sum_{s \in \mathcal{S}} \left(\ln(\sqrt{2\pi}\sigma_{\omega_s}) + \frac{(f_s - \mu_{\omega_s})^2}{2\sigma^2_{\omega_s}} \right) \tag{2.50}$$

$$\text{and } U_2(\omega) = \sum_{C \in \mathcal{C}} V_2(\omega_C) \tag{2.51}$$

$$\text{where } V_2(\omega_C) = V_{\{s,r\}}(\omega_s, \omega_r) = \begin{cases} -\beta & \text{if } \omega_s = \omega_r \\ +\beta & \text{if } \omega_s \neq \omega_r \end{cases} \tag{2.52}$$

where β is a model parameter controlling the homogeneity of the regions. As β increases, the resulting regions become more homogeneous. Clearly, we have $2L + 1$

parameters. They are denoted by the vector Θ:

$$\Theta = \begin{pmatrix} \vartheta_0 \\ \vartheta_1 \\ \vdots \\ \vartheta_{2L} \end{pmatrix} \equiv \begin{pmatrix} \mu_0 \\ \mu_1 \\ \vdots \\ \mu_{L-1} \\ \sigma_0 \\ \vdots \\ \sigma_{L-1} \\ \beta \end{pmatrix} \tag{2.53}$$

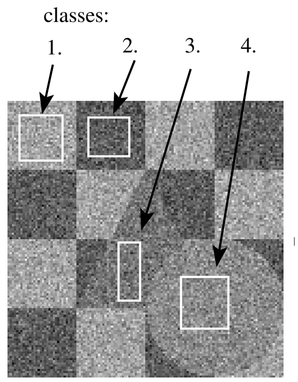

Figure 2.3: *Training sets on a synthetic image*

If the parameters are supposed to be known, we say that the segmentation process is *supervised*. If they are unknown (and hence they have to be estimated simultaneously during the segmentation), the segmentation process is called *unsupervised*. Unsupervised segmentation and parameter estimation methods will be discussed later in Chapter 4.

For supervised segmentation, we are given a set of training data (small sub-images), each of them representing a class (see Figure 2.3). According to the law of large numbers (see Section 1.5), we approach the statistics of the classes (mean and variance) by the *empirical mean* and *empirical variance*:

$$\forall \lambda \in \Lambda : \qquad \mu_\lambda = \frac{1}{\mid S_\lambda \mid} \sum_{s \in S_\lambda} f_s, \tag{2.54}$$

$$\sigma_\lambda^2 = \frac{1}{\mid S_\lambda \mid} \sum_{s \in S_\lambda} (f_s - \mu_\lambda)^2, \tag{2.55}$$

where S_λ is the set of pixels included in the training set of class λ. The parameter β is initialized in an ad-hoc way (by trial and error). Typical values are between 0.5 and 1.

In Figure 2.2, we give an overview of a supervised segmentation process. We have two inputs: the image itself and the parameters Θ. They yield an energy function as defined in Equation (2.50)–Equation (2.52). To find the MAP estimate, an algorithm is needed to minimize this energy function. In Chapter 3, we discuss a variety of such algorithms. Here, we have used the Gibbs Sampler [46] to get the minimum. The resulting image is just the labeling with minimum energy.

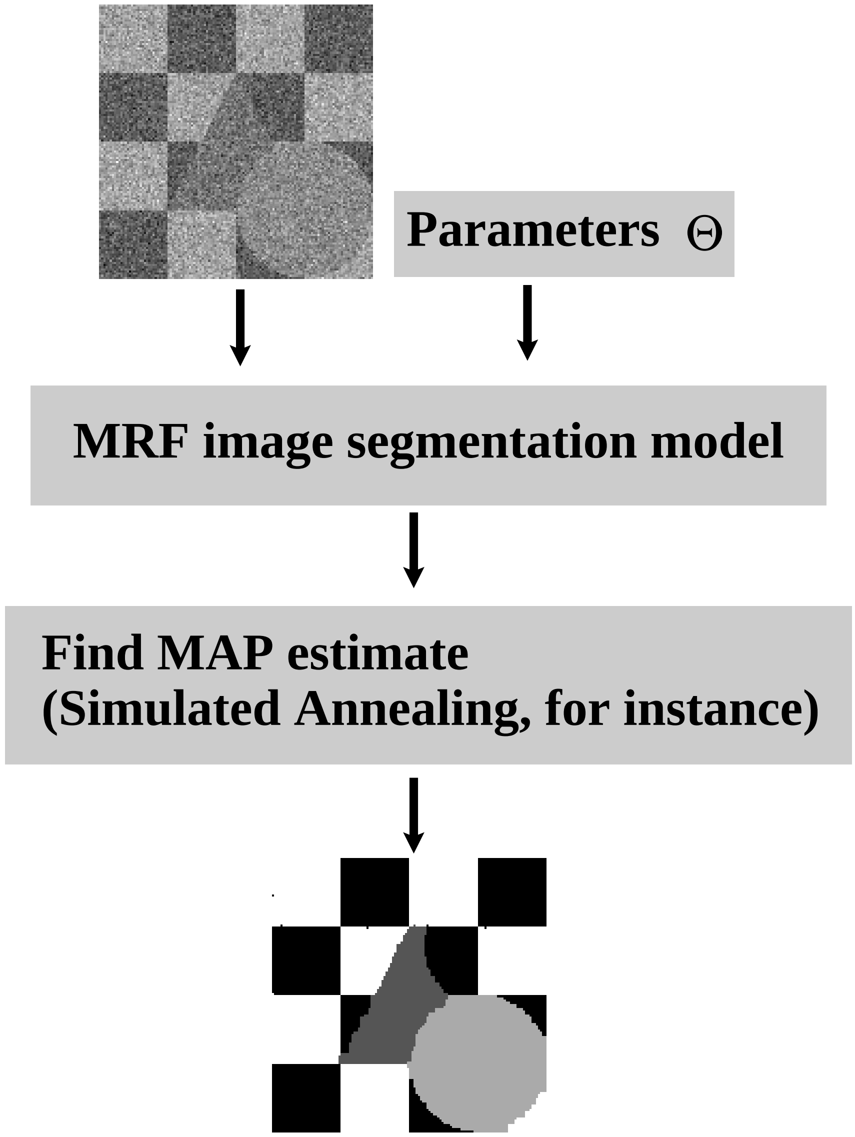

Figure 2.2: *Supervised image segmentation process*

2.4 Multigrid Approaches

Early vision (or low level vision) processes deal with massive amounts of data. Thus such algorithms have two requirements in order to accomplish their tasks: they should be highly parallel to handle the data in a short time and they should provide a structure simplifying the extraction of higher level image features such as edges, regions, etc... Parallel multigrid (or pyramidal) schemes are one of the possible approaches satisfying these demands. A recent book [71] on the subject is discussing various aspects of multiresolution image processing. Computation on image data with pyramids has importance not only in vision but also in the development of parallel computers (for some examples, see [71]). From a biological point of view, they support algorithms that share properties with the human vision (the human retina acquires visual information at multiple resolution at the same time).

Multigrid methods have a long existence in numerical analysis (partial differential equations, for instance). In image processing, they have also been used in various contexts from the mid 70's. Many vision problems have been formulated in terms of the optimization of a cost function. The multigrid approach of such an optimization is similar to the one used in numerical analysis (see [97] for a discussion about). Here, we are interested in pyramidal methods applied to MRF image modelization [55]. We use the notion *pyramidal* to designate *multigrid* and *hierarchical* schemes. We are talking about *multigrid* methods, if the layers in the pyramid are not connected. In this case, the optimization algorithm is usually parallelizable only on the layers, but it is still sequential between layers. The layout of a *multigrid* model can be represented by a stack of smaller and smaller image lattices. If there is an inter-level communication, the model is called *hierarchical*. While the optimization algorithms associated with such models can be parallelized on the whole pyramid, the underlying MRF model becomes more complicated requiring more computation. The layout of the model is represented by a tree.

As we explained in Section 2.2, we usually have two processes in a MRF model: the *observation process* and the *label process*. In a multigrid scheme, we usually build scales with different resolution using the *label process* and keep the whole *observation process* [88, 16, 50, 17, 64, 18, 85, 99]. Herein, we briefly review some related techniques.

2.4.1 Renormalization Group Approach

Based on renormalization group ideas from statistical physics, this technique has been adapted by B. Gidas [50] to image processing. The main advantage of the method is that it provides a mechanism for relating the processing at different scales with one another. This mechanism is a nonlinear transformation—called the *Renormalization Group (RG) transformation*. Following [50], let us briefly review the Renormalization Group Algorithm (RGA):

The RGA consists of two major steps: a *renormalization step* and a *processing step*. In the former step, we iteratively build a sequence of coarser and coarser grids

$$\mathcal{S} \equiv \mathcal{S}^0 \to \mathcal{S}^1 \to \mathcal{S}^2 \to \ldots \to \mathcal{S}^M \tag{2.56}$$

and a corresponding sequence of energy functions

$$U^0 \to U^1 \to \ldots \to U^M. \tag{2.57}$$

The coarsening of the grid $\mathcal{S}^0$ is done by dividing it into blocks of $w \times h$ (assuming that $\mathcal{S}^0$ is a lattice of size $w^M \times h^M$) and then identifying each block by one of its elements. The collection of these sites will form $\mathcal{S}^1$. Proceeding in the same way, we obtain the other grids. The Hamiltonians are obtained from the original energy U iteratively as follows: for each $m, 1 \leq m \leq M$, we choose a conditional probability

$$P^m(\mathcal{X}^m = \omega^m \mid \mathcal{X}^{m-1} = \omega^{m-1}) \tag{2.58}$$

which expresses the probability that the label field at $\mathcal{S}^m$ is ω^m given that the configuration at $\mathcal{S}^{m-1}$ is ω^{m-1}. The energy function of coarser grids are computed by the next formula:

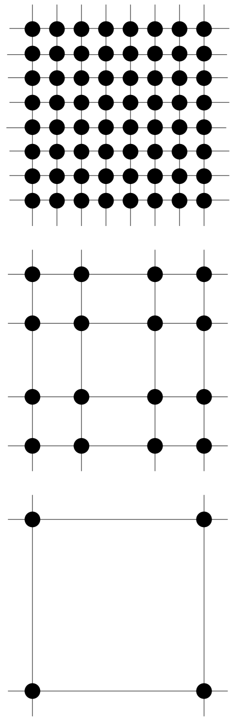

Figure 2.4: *Renormalization group approach.*

$$\forall m \colon m = 1, \ldots, M :$$

$$U_K^m(\omega^m) = -\ln\left(\sum_{\mathcal{S}} P^m(\omega^m \mid \omega^{m-1}) \exp(-U_K^{m-1}(\omega^{m-1}))\right), \tag{2.59}$$

where K is a positive constant and $U_K^0 = \frac{1}{K}U$. The above equation could be rewritten as

$$\forall m \colon m = 1, \ldots, M : \quad U_K^m = R^m(U_K^{m-1}). \tag{2.60}$$

R^m is the *RG transformation* at stage m. We remark that other RG transformations might also be defined (see [50] for more details). Now, we organize the grids into a *vertical cascade* with the finest grid at the top as shown in Figure 2.4. The *processing step* of RGA is essentially a multiscale, coarse-to-fine processing starting at the bottom level: First we solve the problem at the coarsest level then we move upwards transmitting the obtained information at level m to the level $m-1$ above. At level $m-1$, the processing is done under a constraint introduced by the level m below. Let us examine this technique in the case of the Hamiltonians defined in Equation (2.59). First, we find the global minimum of U_K^m, then we are looking for the global minimum $\hat{\omega}^{m-1}$ of U_K^{m-1} but searching *only* in a smaller subspace of configurations constrained by the minimum $\hat{\omega}^m$ of U_K^m via the equation

$$P^m(\hat{\omega}^m \mid \omega^{m-1}) = \max_{\omega^m} P^m(\omega^m \mid \omega^{m-1}). \tag{2.61}$$

Obviously, this reduces the computational complexity of each level. In [50], it has been shown that if the above probabilities are properly chosen and if K is sufficiently small, RGA computes the correct MAP estimate of the original image at level 0. The parameter K could be interpreted as a temperature and thus we can use a Simulated Annealing algorithm for the energy-minimization. Furthermore, the equilibrium properties of the field $m-1$ at temperature K are related to the properties of the field m at temperature one. In some simple problems, however, the use of such a *stochastic relaxation algorithm* is not needed. The general formulation of the RGA is the following [50]:

Algorithm 2.4.1 (Renormalization Group Algorithm)

① *Generate a cascade of coarse grids from the original image, choose a set of conditional probabilities* $P^m, m = 1, \ldots, M$, *and a positive constant* K *sufficiently small and compute the coarse Hamiltonians* $U_K^m, m = 1, \ldots, M$.

② *Set* $m = M$ *and find the global minimum* $\hat{\omega}^M$ *of* U_K^M.

③ *Find the global minimum* $\hat{\omega}^{m-1}$ *of* U_K^{m-1} *among all configurations satisfying Equation (2.61).*

④ *Stop if* $m = 1$, *return to Step ③ with* $m = m - 1$ *otherwise.*

Finally, we give a very simple example of the conditional probabilities P^m mentioned in Step ① of the above algorithm. Suppose that the sites of the grid $\mathcal{S}^m$ are also the sites of the grid $\mathcal{S}^{m-1}$ but not vice versa (as in Figure 2.4, for example). Then we choose

$$P^m(\omega^m \mid \omega^{m-1}) = \prod_{s \in \mathcal{S}^m} \delta(\omega_s^m, \omega_s^{m-1}) \tag{2.62}$$

where $\delta(a, b) = 1$ if $a = b$ and 0 otherwise.

In summary, RGA is a quite consistent approach. The coarser grids and their Hamiltonians are well defined, they are deduced from the original image. Another important property is that RGA finds the MAP estimate of the *original* label field which is not true for all multigrid approaches. It can be applied not only to MRF models but also to any other model with a cost function and an underlying lattice scheme. The major difficulty is the computation of the energy functions at coarser grids. Usually, this computation cannot be done explicitly (except for some simple models such as the Ising model), one has to approximate them. Another drawback is the loss of Markovianity at coarser grids [97] since the coarser energy functions obtained by the RG transformation cannot be decomposed as a sum of clique-potentials. In [50], the Hamiltonians are approximated by a sum of clique-potentials and hence one can use classical relaxation algorithms to minimize the energy at coarser grids. Unfortunately, such approximations are available only for certain simple models, mainly in image restoration [50, 95].

2.4.2 A Causal Hierarchical MRF Model

Another interesting model has been proposed by Bouman *et al.* [17, 18, 16]. His model consists of a label-pyramid where each level is causally dependent on the coarser layer above it. Bouman also defines a new optimization criteria called *Sequential MAP* (SMAP) estimate. Let us briefly review this model.

First, we build a pyramid as shown on Figure 2.5. Each site at a coarse grid corresponds to a group of 2×2 sites at the grid below it. The fundamental assumption of the model is that the sequence of random fields from coarse to fine scale form a *Markov chain*. Denoting the label field at level n by $\mathcal{X}^n$, this relation may be stated as

$$P(\mathcal{X}^n = \omega^n \mid \mathcal{X}^l = \omega^l, l > n) = P(\mathcal{X}^n = \omega^n \mid \mathcal{X}^{n+1} = \omega^{n+1}). \tag{2.63}$$

The observation field $\mathcal{F}$ depends only on the labeling at the finest scale implying

$$P(\mathcal{F} = f \mid \mathcal{X}^n, n > 0) = P(\mathcal{F} = f \mid \mathcal{X}^0). \tag{2.64}$$

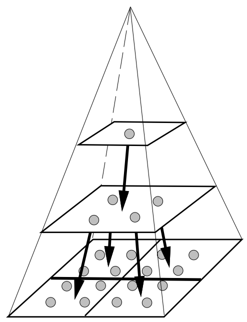

Figure 2.5: *A causal hierarchical model.*

From the above equations, we can easily deduce the joint distribution of $\mathcal{F}$ and $\mathcal{X}$:

$$P(\mathcal{F} = f \mid \mathcal{X} = \omega) = P(f \mid \omega^0) \left(\prod_{n=0}^{M-1} P(\omega^n \mid \omega^{n+1}) \right) P(\omega^M). \tag{2.65}$$

where M is the coarsest scale.

As pointed out in [16], the conventional MAP estimate is not satisfying since its cost function (cf. Equation (2.15)) would assigns equal cost to a single mislabeled pixel at $n = 0$ or to the mislabeling of hundreds of pixels at the coarsest scale. The proposed solution is to define a cost function related to the width of the largest grouping of mislabeled pixels. More precisely, let K be the coarsest scale containing a misclassified pixel. Obviously, the error at scale K will influence the labeling at finer scales leading to the misclassification of a group of pixels at the finest scale. The width of this group will be approximately 2^K. The new cost function is of the following form:

$$\begin{aligned} C_{SMAP}(\mathcal{X}, \omega) &= \frac{1}{2} + \sum_{n=0}^{M} 2^{n-1} C_n(\mathcal{X}, \omega) && (2.66) \\ \text{with } C_n(\mathcal{X}, \omega) &= 1 - \prod_{i=n}^{M} \delta(\mathcal{X}^i, \omega^i), && (2.67) \end{aligned}$$

where δ is the Dirac mass. The estimate $\hat{\omega}$ of the label field is obtained by minimizing the risk:

$$\begin{aligned} \hat{\omega} &= \arg\min_{\omega} E\{C_{SMAP}\} && (2.68) \\ &= \arg\min_{\omega} \sum_{n=0}^{M} 2^{n-1}(1 - P(\omega^i, i \geq n \mid f)) && (2.69) \\ &= \arg\max_{\omega} \sum_{n=0}^{M} 2^{n} P(\omega^i, i \geq n \mid f) && (2.70) \end{aligned}$$

This estimate can be computed recursively since the fields $\mathcal{X}^n$ form a Markov chain. Denoting the estimate obtained at level i by $\hat{\omega}^i$, we obtain the next procedure:

$$\begin{aligned} \hat{\omega}^M &= \arg\max_{\omega^M} \ln(P(\omega^M \mid f)) && (2.71) \\ &= \arg\max_{\omega^M} \ln(P(f \mid \omega^M)) \text{ (assuming that } \mathcal{X}^M \text{ is uniform)} && (2.72) \\ \hat{\omega}^n &= \arg\max_{\omega^n} \ln(P(\omega^n \mid \hat{\omega}^{n+1}, f)) && (2.73) \\ &= \arg\max_{\omega^n} (\ln(P(f \mid \omega^n)) + \ln(P(\omega^n \mid \hat{\omega}^{n+1}))), \quad n = M-1, \ldots, 0 && (2.74) \end{aligned}$$

The procedure is initialized by the MAP estimate of the coarsest scale given the observed data. At finer scales, we are looking for the MAP estimate of $\mathcal{X}^n$, given the observations *and* the estimate $\hat{\omega}^{n+1}$ at the scale above it. Due to this structure, this estimator is called *sequential MAP* (SMAP).

Algorithm 2.4.2 (Causal Hierarchical Algorithm)

① *Build a pyramid from the label field $\mathcal{X}$ by dividing the original grid into coarser scales. Each site corresponds to a block of 2×2 sites at the level below it.*

② *Find the optimal labeling at the coarsest scale using Equation (2.72) and set $n = M - 1$.*

③ *Find the MAP estimate of the label field at scale n given the observations f and the estimate ω^{n+1} at the coarser level above it using Equation (2.74).*

④ *Stop if $n = 0$, go to Step ③ with $n = n - 1$ otherwise.*

The SMAP estimator has many advantages. The most important is that it can be obtained with a single non-iterative pass in contrast to the MAP or MPM estimates (for more details, see [16]).

2.5 A Multiscale MRF Model

This multiscale model has been proposed by Perez *et al.* in [64, 63] for motion analysis using a second order neighborhood system shown in Figure 1.10 (see [97] for a more general description of the model). Herein, we give a general description of this model. Then we study it in the case of a first order neighborhood system (see Figure 1.9) which is the most commonly used in image segmentation problems.

2.5.1 General Description

Let us suppose that $\mathcal{S} = \{s_1, s_2, \ldots, s_N\}$ is a $W \times H$ lattice, so that:

$$\mathcal{S} \equiv \mathcal{L} = \{(i,j) : 1 \leq i \leq W \text{ and } 1 \leq j \leq H\}, \tag{2.75}$$

and[3] $W = w^n$, $H = h^m$. Furthermore, we have some neighborhood system $\mathcal{V}$ on these sites. Let $\mathcal{X}$ be a MRF over $\mathcal{V}$ with an energy function U and potentials $\{V_C\}_{C \in \mathcal{C}}$. The following procedure will generate the multigrid MRF corresponding to $\mathcal{X}$:

1. Let $\mathcal{B}^0 \equiv \mathcal{S}$ and $\Omega_0 \equiv \Omega$.

[3] This assumption introduces some restrictions on $\mathcal{L}$ but this is not crucial in practice since we work mostly on images where both W and H is a power of 2.

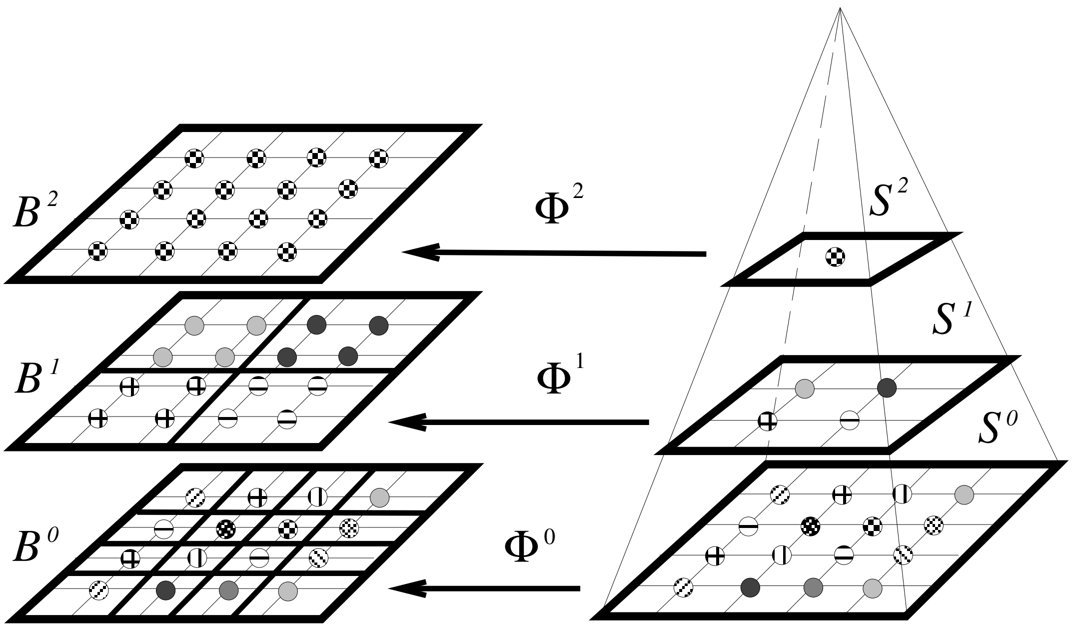

Figure 2.6: *The isomorphism Φ^i between $\mathcal{B}^i$ and $\mathcal{S}^i$.*

2. For all $1 \leq i \leq M$ ($M = \inf(n, m)$), $\mathcal{S}$ is divided into blocks of size $w^i \times h^i$. These blocks will form the scale $\mathcal{B}^i = \{b_1^i, \ldots, b_{N_i}^i\}$ ($N_i = N/(wh)^i$).

The labels assigned to the sites of a block are supposed to be the same over the whole block. The common label of the block b_k^i is denoted by $\omega_k^i \in \Lambda$. This constraint yields a configuration space Ω_i which is a subset of the original set Ω. Obviously, for all $0 \leq i \leq M$: $\Omega_i \subset \Omega_{i-1} \subset \cdots \subset \Omega_0 \equiv \Omega$.

Now, let us consider the neighborhood system at scale i. It is clear, that b_k^i and b_l^i are neighbors if and only if there exist two neighbors $s \in \mathcal{S}$ and $r \in \mathcal{S}$ such that $s \in b_k^i$ and $r \in b_l^i$. This yields the same cliques as in $\mathcal{C}$. The cliques can be defined in the following way: Let $d = \deg(\mathcal{C})$. For all $1 \leq j \leq d$, the set of j blocks C_j^i at scale i is a clique of order j if there exists a clique $C \in \mathcal{C}$ (that is a clique at the finest scale) such that:

$$1. \quad C \subseteq \bigcup_{b_k^i \in C_j^i} b_k^i$$

$$2. \quad \forall b_k^i \in C_j^i : C \cap b_k^i \neq \emptyset.$$

The set of cliques at scale i is denoted by $\mathcal{C}^i$ ($\mathcal{C}^0 \equiv \mathcal{C}$). The set of all cliques satisfying 1 and 2 for a given C_j^i is denoted by $\mathcal{D}_{C_j^i} \subseteq \mathcal{C}$.

Let us partition the original set $\mathcal{C}$ into the following disjoint subsets: For all $1 \leq j \leq d$, let $\mathcal{A}^i_j$ be the set of cliques $C \in \mathcal{C}$ for which there exists a clique C^i_j (that is a clique of order j at the scale i) satisfying 1 and 2. Then, it turns out from the definition of $\mathcal{D}_{C^i_j}$ and $\mathcal{A}^i_j$, that

$$\mathcal{A}^i_j = \bigcup_{C^i_j \in \mathcal{C}^i} \mathcal{D}_{C^i_j}. \tag{2.76}$$

Using this partition, the energy function U can be decomposed in the following way:

$$\begin{aligned} U(\omega) = \sum_{C \in \mathcal{C}} V_C(\omega) &= \sum_{C \in \mathcal{A}^i_1} V_C(\omega) + \cdots + \sum_{C \in \mathcal{A}^i_d} V_C(\omega) & (2.77) \\ &= \sum_{C^i_1 \in \mathcal{C}^i} \sum_{C \in \mathcal{D}_{C^i_1}} V_C(\omega) + \cdots + \sum_{C^i_d \in \mathcal{C}^i} \sum_{C \in \mathcal{D}_{C^i_d}} V_C(\omega) & (2.78) \end{aligned}$$

The main benefit of this decomposition is that the potentials at coarser scales can be derived by simple computation from the potentials at the finest scale. If we note the potential corresponding to a clique C^i_j of order j at the scale i by $V^{\mathcal{B}^i}_{C^i_j}$, we have the following family of potentials at scale $\mathcal{B}^i$:

$$V^{\mathcal{B}^i}_{C^i_j}(\omega) = \sum_{C \in \mathcal{D}_{C^i_j}} V_C(\omega) \tag{2.79}$$

If we examine our model, we see that there is some redundancy at coarser scales: we have the same label over the sites of a block. It seems then natural to associate a unique site to each block. These sites have the common state of the corresponding block and they form a coarser grid $\mathcal{S}^i$ isomorphic to the corresponding scale $\mathcal{B}^i$. The coarser configuration space $\Xi_i = \{\xi^i_s : s \in \mathcal{S}^i, \xi^i_s \in \Lambda\}$ is isomorphic to Ω_i. Obviously, $\Xi_0 \equiv \Omega_0 \equiv \Omega$. The isomorphism Φ^i from $\mathcal{S}^i$ in $\mathcal{B}^i$ is just a projection of the coarser label field to the fine grid $\mathcal{S}^0 \equiv \mathcal{S}$:

$$\begin{aligned} \Phi^i \ : \ & \Xi_i \longrightarrow \Omega_i \\ & \xi^i \longmapsto \omega = \Phi^i(\xi^i). \end{aligned} \tag{2.80}$$

Φ^i keeps the same neighborhood structure on $\mathcal{S}^i$ as on $\mathcal{B}^i$ and the cliques on $\mathcal{S}^i$ inherit the potentials from the cliques defined on $\mathcal{B}^i$. These grids form a pyramid where level i contains the grid $\mathcal{S}^i$. The energy function of level i ($i = 0, \ldots, M$) is of the form:

$$\begin{aligned} U^i(\xi^i) &= \sum_{C^i \in \mathcal{C}^i} V^i_{C^i}(\xi^i) \quad i = 0, \ldots, M & (2.81) \\ \text{where } V^i_{C^i}(\xi^i) &= V^{\mathcal{B}^i}_{C^i}(\Phi^i(\xi^i)). & (2.82) \end{aligned}$$

The multiscale algorithm essentially follows a top-down strategy (see Figure 2.8). First the highest layer of the pyramid is solved, then the next level is initialized by the result. The general formulation of the multiscale algorithm is the following:

Algorithm 2.5.1 (Multiscale MRF Algorithm)

① *Let* $\mathcal{B}^0 \equiv \mathcal{S}$, $\Omega_0 \equiv \Omega$ *and divide* $\mathcal{S}$ *into blocks of size* $w^i \times h^i$ $(1 \le i \le M)$. *Then associate a unique site to each block forming a coarse grid.*

② *Compute the clique-potentials at coarse grids using Equation (2.82).*

③ *Set* $i = M$ *and find the global minimum* $\hat{\xi}^M$ *of* U^i *in Equation (2.81).*

④ *Initialize the layer* $i-1$ *by a projection of* $\hat{\xi}^i$ *into* $\mathcal{S}^{i-1}$: $\xi^{i-1} = (\Phi^{i-1})^{-1} \circ \Phi^i(\hat{\xi}^i)$, *and find the minimum* $\hat{\xi}^{i-1}$ *of* U^{i-1}.

⑤ *Stop if* $i = 1$, *return to Step* ④ *with* $i = i - 1$ *otherwise.*

The advantages of this algorithm are clear: each $\hat{\xi}^i$ gives a more or less good estimate of the final result. The estimate is better as i goes down to 0. On the other hand, for the higher values of i, the corresponding problem is simpler since the state space has only a few elements.

This scheme is particularly well adapted to the deterministic relaxation methods which are more sensitive to the initial configuration than the stochastic[4] ones.

Another advantage is that the method keeps the Markov property at coarse grids which is not true for the RGA presented in Section 2.4.1.

2.5.2 A Special Case

In the following, we will focus on a MRF with a first order neighborhood-system (see Figure 1.9) where the energy function is given by:

$$U(\omega, \mathcal{F}) = U_1(\omega, \mathcal{F}) + U_2(\omega). \tag{2.83}$$

U_1 (resp. U_2) denotes the energy of the first order (resp. second order) cliques. The notation $U_1(\omega, \mathcal{F})$ means that the first order potentials depend not only on the actual labeling but also on the given observations.

[4] Deterministic and stochastic relaxation algorithms will be discussed later in Chapter 3. Here, we only note that they are used to find a minimum of a non-convex energy function. Deterministic algorithms are usually faster than stochastic ones but they depend on the initial conditions. Stochastic algorithms find a global optimum starting from any initial configuration but they are much slower.

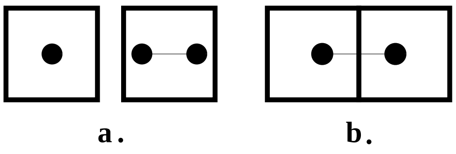

Figure 2.7: *The two subsets of $\mathcal{C}$ in the case of a first order neighborhood system. a: $\mathcal{C}_k^i$; b: $\mathcal{C}_{k,l}^i$.*

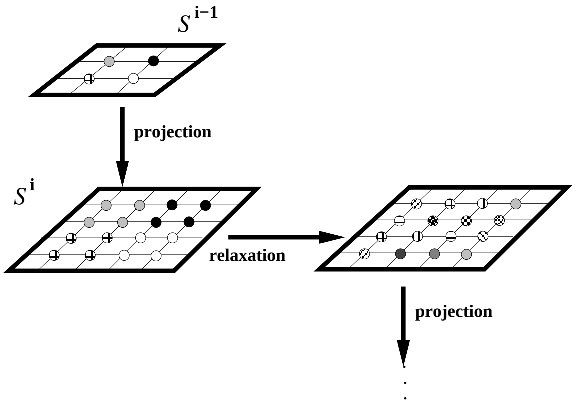

Figure 2.8: *Multiscale relaxation scheme.*

We follow the procedure described in Section 2.5.1 to generate a multigrid MRF model. Let $\mathcal{B}^i = \{b_1^i, \ldots, b_{N_i}^i\}$ denote the set of blocks and Ω_i the configuration-space at scale i ($\Omega_i \subset \Omega_{i-1} \subset \cdots \subset \Omega_0 = \Omega$). The label associated with block b_k^i is denoted by ω_k^i. We can define the same neighborhood structure on $\mathcal{B}^i$ as on $\mathcal{S}$:

$$b_k^i \text{ and } b_l^i \text{ are neighbors} \Longleftrightarrow \begin{cases} b_k^i \equiv b_l^i \text{ or} \\ \exists C \in \mathcal{C} \mid C \cap b_k^i \neq \emptyset \text{ and } C \cap b_l^i \neq \emptyset \end{cases} \tag{2.84}$$

Now, let us partition the original set $\mathcal{C}$ into two disjoint subsets $\{\mathcal{C}_k^i\}$ and $\{\mathcal{C}_{k,l}^i\}$:

1. cliques which are included in b_k^i (see Figure 2.7/a.):

$$\mathcal{C}_k^i = \{C \in \mathcal{C} \mid C \subset b_k^i\} \tag{2.85}$$

2. cliques which sit astride two neighboring blocks $\{b_k^i, b_l^i\}$ (see Figure 2.7/b.):

$$\mathcal{C}_{k,l}^i = \{C \in \mathcal{C} \mid C \subset (b_k^i \cup b_l^i) \text{ and } C \cap b_k^i \neq \emptyset \text{ and } C \cap b_l^i \neq \emptyset\} \tag{2.86}$$

It is obvious from this partition that our energy function (see Equation (2.83)) can be decomposed as:

$$U_1(\omega, \mathcal{F}) \quad = \quad \sum_{s \in \mathcal{S}} V_1(\omega_s, f_s)$$

$$
\begin{aligned}
&= \sum_{b_k^i \in \mathcal{B}^i} \underbrace{\sum_{s \in b_k^i} V_1(\omega_s, f_s)}_{V_1^{\mathcal{B}^i}(\omega_k^i, \mathcal{F})} = \sum_{b_k^i \in \mathcal{B}^i} V_1^{\mathcal{B}^i}(\omega_k^i, \mathcal{F}) && (2.87)\\
\text{and } U_2(\omega) &= \sum_{C \in \mathcal{C}} V_2(\omega_c) \\
&= \sum_{b_k^i \in \mathcal{B}^i} \underbrace{\sum_{C \in \mathcal{C}_k^i} V_2(\omega_c)}_{V_k^{\mathcal{B}^i}(\omega_k^i)} + \sum_{\{b_k, b_l\} neighbors} \underbrace{\sum_{C \in \mathcal{C}_{k,l}^i} V_2(\omega_c)}_{V_{k,l}^{\mathcal{B}^i}(\omega_k^i, \omega_l^i)} \\
&= \sum_{b_k^i \in \mathcal{B}^i} V_k^{\mathcal{B}^i}(\omega_k^i) + \sum_{\{b_k, b_l\} neighbors} V_{k,l}^{\mathcal{B}^i}(\omega_k^i, \omega_l^i) && (2.88)
\end{aligned}
$$

Now, we can define our pyramid (cf. Figure 2.6) where level i contains the coarse grid $\mathcal{S}^i$ which is isomorphic to the scale $\mathcal{B}^i$. The coarse grid has a reduced configuration space $\Xi_i = \Lambda^{N_i}$.

The model on the grids $\mathcal{S}^i$ $(i = 0, \ldots, M)$ defines a set of consistent multiscale MRF models, whose energy functions are derived from Equations (2.87) and (2.88)

$$
\begin{aligned}
U^i(\xi^i, \mathcal{F}) &= U_1^i(\xi^i, \mathcal{F}) + U_2^i(\xi^i) && (2.89)\\
&= U_1(\Phi^i(\xi_i), \mathcal{F}) + U_2(\Phi^i(\xi_i)) \quad i = 0, \ldots, M \\
\text{with } U_1^i(\xi^i, \mathcal{F}) &= \sum_{k \in \mathcal{S}^i} (V_1^{\mathcal{B}^i}(\omega_k^i, \mathcal{F}) + V_k^{\mathcal{B}^i}(\omega_k^i)) = \sum_{k \in \mathcal{S}^i} V_1^i(\xi_k^i, \mathcal{F}) && (2.90)\\
\text{and } U_2^i(\xi^i) &= \sum_{\{k,l\} neighbors} V_{k,l}^{\mathcal{B}^i}(\omega_k^i, \omega_l^i) = \sum_{C^i \in \mathcal{C}^i} V_2^i(\xi_C^i) && (2.91)
\end{aligned}
$$

where C^i is a second order clique corresponding to the definition in Equation (2.84) and $\mathcal{C}^i$ is the set of cliques on the grid $\mathcal{S}^i$.

2.5.3 Application to Image Segmentation

We can easily apply the equations obtained at the previous section to the image segmentation model presented in Section 2.3. For simplicity, the block size is supposed to be $n \times n$ (that is $w = h = n$). Then, we get [74, 75, 72]:

$$
\begin{aligned}
U_1^i(\xi^i, \mathcal{F}) &= \sum_{s^i \in \mathcal{S}^i} V_1^i(\xi_{s^i}^i, \mathcal{F}) \\
\text{where } V_1^i(\xi_{s^i}^i, \mathcal{F}) &= \sum_{s \in b_{s^i}^i} V_1(\omega_s, f_s) + \sum_{C \in \mathcal{C}_{s^i}^i} V_2(\omega_C)
\end{aligned}
$$

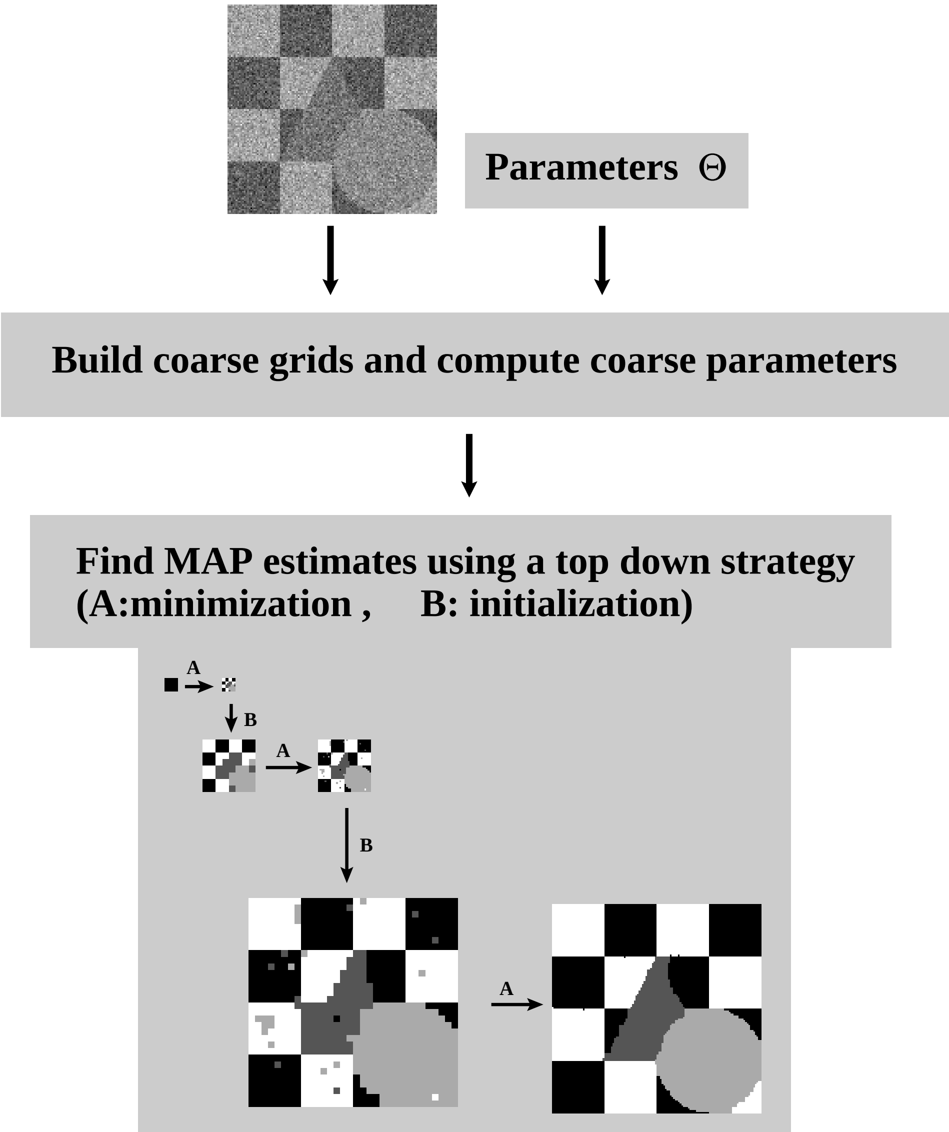

Figure 2.9: *Multiscale supervised image segmentation process.*

$$= \sum_{s \in b^i_{s^i}} \left(\log(\sqrt{2\pi}\sigma_{\omega_s}) + \frac{(f_s - \mu_{\omega_s})^2}{2\sigma^2_{\omega_s}} \right) - p^i\beta \tag{2.92}$$

$$\text{and } U_2^i(\xi^i) = \sum_{C^i = \{r^i, s^i\} \in \mathcal{C}^i} V_2^i(\xi^i_{C^i})$$

$$\text{where } V_2^i(\xi^i_{C^i}) = \sum_{\{r,s\} \in \mathcal{D}_{C^i}} V_2(\omega_r, \omega_s) = \begin{cases} -q^i\beta & \text{if } \omega_r = \omega_s \\ +q^i\beta & \text{if } \omega_r \neq \omega_s \end{cases} \tag{2.93}$$

The values of p^i and q^i depend on the chosen block size and the neighborhood structure. p^i is the number of cliques included in the same block at scale $\mathcal{B}^i$ and q^i is the number of cliques between two neighboring blocks at scale $\mathcal{B}^i$. Considering blocks of $n \times n$ and a first order neighborhood system, we get:

$$p^i = 2n^i(n^i - 1) \tag{2.94}$$

$$q^i = n^i \tag{2.95}$$

In Figure 2.9, we give an overview of a *supervised* multiscale segmentation process. As in the monogrid case, we have two inputs: the image and the monogrid parameters $\Theta \equiv \Theta^0$ defined in Equation (2.53). Then, we build the pyramid and compute the parameters $\Theta^i (i = 1, \ldots, M)$ at coarse grids obtaining $M + 1$ energy functions defined by Equation (2.92)–Equation (2.93). Using a *top-down* strategy, we minimize these functions (using the Iterated Conditional Mode algorithm [13], for example) and we take the labeling at the finest level as the final segmentation.

2.6 The Hierarchical Model

In this section, we propose a new hierarchical MRF model [73, 76, 74, 75, 72]. The basic idea is to find a better way of communication between the levels than the initialization used for the multiscale model. Our approach consists in introducing new interactions between two neighbor grids[5] in the pyramid. This scheme permits also the parallelization of the relaxation algorithm on the whole pyramid. First, we give a general description of the model, then we study a special case with a first order neighborhood system.

2.6.1 General Description

[5]One can imagine interactions between more than two levels but these schemes are too complicated for practical use.

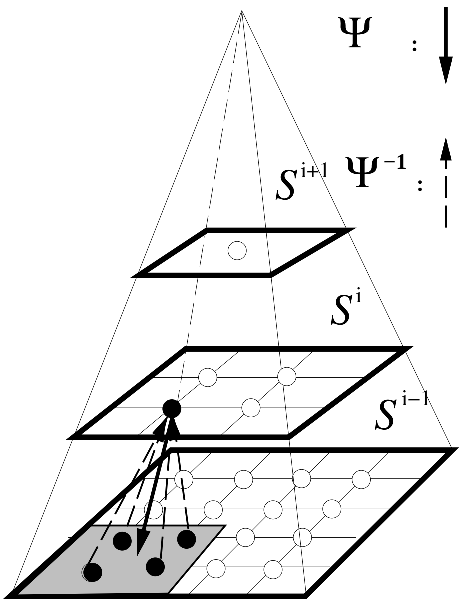

Figure 2.10: *The functions* Ψ *and* Ψ^{-1}

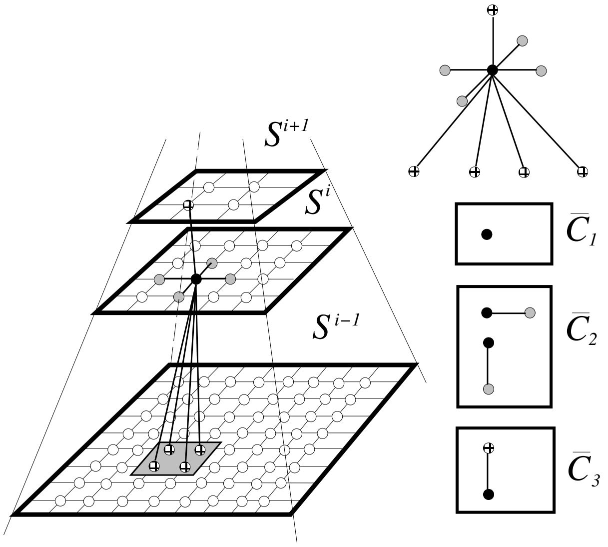

Figure 2.11: *The neighborhood system* $\bar{\mathcal{V}}$ *and the cliques* $\bar{\mathcal{C}}_1$, $\bar{\mathcal{C}}_2$ *and* $\bar{\mathcal{C}}_3$.

We consider hereafter the label pyramid and the whole observation field defined in the previous section. Let $\bar{\mathcal{S}} = \{\bar{s}_1, \ldots, \bar{s}_{\bar{N}}\}$ denote the sites of this pyramid. Obviously,

$$\begin{aligned} \bar{\mathcal{S}} &= \bigcup_{i=0}^{M} \mathcal{S}^i \qquad (2.96) \\ \bar{N} &= \sum_{i=0}^{M} N_i. \end{aligned}$$

$\bar{\Omega}$ denotes the configuration-space of the pyramid:

$$\begin{aligned} \bar{\Omega} &= \Xi^0 \times \Xi^1 \times \cdots \times \Xi^M \\ &= \{\bar{\omega} \mid \bar{\omega} = (\xi^0, \xi^1, \ldots, \xi^M)\} \qquad (2.97) \end{aligned}$$

Let us define the following function Ψ between two neighbor levels, which assigns to a site of any level the corresponding block of sites at the level below it (that is its descendants). Ψ^{-1} assigns its ancestor to a site (see Figure 2.10):

$$\begin{aligned} \Psi : & \quad \mathcal{S}^i \longrightarrow \mathcal{S}^{i-1} \\ \Psi(\bar{s}) &= \{\bar{r} \mid \bar{s} \in \mathcal{S}^i \Rightarrow \bar{r} \in \mathcal{S}^{i-1} \text{ and } b_{\bar{r}}^{i-1} \subset b_{\bar{s}}^i\} \qquad (2.98) \end{aligned}$$

Now, we can define on these sites the following neighborhood-system (see Figure 2.11):

$$\bar{\mathcal{V}} = (\bigcup_{i=0}^{M} \mathcal{V}_i) \cup \{\Psi^{-1}(\bar{s}) \cup \Psi(\bar{s}) \mid \bar{s} \in \bar{\mathcal{S}}\} \tag{2.99}$$

where $\mathcal{V}_i$ is the neighborhood structure of the i^{th} level, and we have the following cliques:

$$\bar{\mathcal{C}} = (\bigcup_{i=0}^{M} \mathcal{C}^i) \cup \mathcal{C}^* \tag{2.100}$$

where $\mathcal{C}^*$ denotes the new cliques siting astride two neighbor grids. We can easily estimate the degree of the new cliques since it depends on the block size: Each site interacts with its ancestor (there is one) and its descendants (there are wh), thus:

$$\begin{aligned} \deg(\mathcal{C}^*) &= \max_{C^* \in \mathcal{C}^*} \mid C^* \mid = wh + 2 && (2.101)\\ \text{and } \deg(\bar{\mathcal{C}}) &= \deg(\mathcal{C}) + \deg(\mathcal{C}^*) - 1. && (2.102) \end{aligned}$$

Furthermore, let $\bar{\mathcal{X}}$ be a MRF over $\bar{\mathcal{V}}$ with an energy function $\bar{U}$ and potentials $\{\bar{V}_{\bar{C}}\}_{\bar{C} \in \bar{\mathcal{C}}}$. The energy function is of the following form:

$$\begin{aligned} \bar{U}(\bar{\omega}) &= \sum_{\bar{C} \in \bar{\mathcal{C}}} \bar{V}_{\bar{C}}(\bar{\omega}) \\ &= \sum_{i=0}^{M} \sum_{\bar{C} \in \mathcal{C}^i} V^i_{\bar{C}}(\bar{\omega}) + \sum_{\bar{C} \in \mathcal{C}^*} \bar{V}_{\bar{C}}(\bar{\omega}) \\ &= \sum_{i=0}^{M} \sum_{C^i \in \mathcal{C}^i} V^i_{C^i}(\xi^i) + \sum_{C^* \in \mathcal{C}^*} \bar{V}_{C^*}(\bar{\omega}) \\ &= \sum_{i=0}^{M} U^i(\xi^i) + U^*(\bar{\omega}) && (2.103) \end{aligned}$$

It turns out from the above equation, that the energy function consists of two terms. The first one corresponds to the sum of the energy functions of the grids defined in the previous section and the second one ($U^*(\bar{\omega})$) is the energy over the new cliques located between neighbor grids.

Since we have defined a MRF on the whole pyramid, the MAP estimate of the label field is obtained by minimizing the Hamiltonian defined in Equation (2.103). The algorithms are essentially the same as in the monogrid case but in the parallelization, we can take benefit of the pyramidal structure and define a new annealing technique as we will see later in Chapter 3. The final result is the labeling obtained at the finest level as in the multiscale method. However an important difference is that the resulting labeling is not related to the MAP estimate of the finest level (without the pyramid). Intuitively, it can be seen as the MAP estimate of an MRF model (defined on the finest scale) with larger neighborhoods.

2.6.2 A Special Case

In this section, we study the model in the case of a first order neighborhood system. We will consider herein only first and second order cliques. Clique-potentials for the other cliques are supposed to be 0. The cliques can be partitioned into three disjoint subsets $\bar{\mathcal{C}}_1, \bar{\mathcal{C}}_2, \bar{\mathcal{C}}_3$ corresponding to first order cliques, second order cliques which are on the same level and second order cliques which sit astride two neighboring levels (see Figure 2.11). Using this partition, we can derive the following energy function:

$$\begin{aligned}
\bar{U}(\bar{\omega}, \mathcal{F}) &= \bar{U}_1(\bar{\omega}, \mathcal{F}) + \bar{U}_2(\bar{\omega}) && (2.104)\\
\bar{U}_1(\bar{\omega}, \mathcal{F}) &= \sum_{\bar{s}\in\bar{\mathcal{S}}} \bar{V}_1(\bar{\omega}_{\bar{s}}, \mathcal{F})\\
&= \sum_{i=0}^{M} \sum_{s^i\in\mathcal{S}^i} V_1^i(\xi^i_{s^i}, \mathcal{F}) = \sum_{i=0}^{M} U_1^i(\xi^i, \mathcal{F}) && (2.105)\\
\bar{U}_2(\bar{\omega}) &= \sum_{C\in\bar{\mathcal{C}}_2} \bar{V}_2(\bar{\omega}_C) + \sum_{C\in\bar{\mathcal{C}}_3} \bar{V}_2(\bar{\omega}_C)\\
&= \sum_{i=0}^{M} \sum_{C\in\mathcal{C}^i} V_2^i(\xi^i_C) + \sum_{C\in\bar{\mathcal{C}}_3} \bar{V}_2(\bar{\omega}_C)\\
&= \sum_{i=0}^{M} U_2^i(\xi^i) + \sum_{C\in\bar{\mathcal{C}}_3} \bar{V}_2(\bar{\omega}_C) && (2.106)
\end{aligned}$$

2.6.3 Complexity

In this section, we study the complexity of the optimization of the hierarchical model in terms of the required memory (or number of processors in the parallel SIMD implementation) and the required communication compared to the monogrid model.

Memory/processor Let us suppose that our image is of size $W \times H$. Following the procedure described in Section 2.6.1, we generate a pyramid containing $M + 1$ levels. Without loss of generality, we can assume that $W/w \leq H/h$, where $w \times h$ is the block size, both w and h are greater than or equal to two. The hierarchical model requires

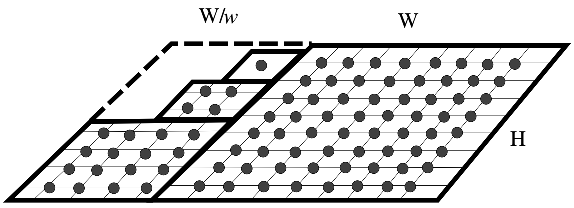

Figure 2.12: *Memory complexity of the hierarchical model*

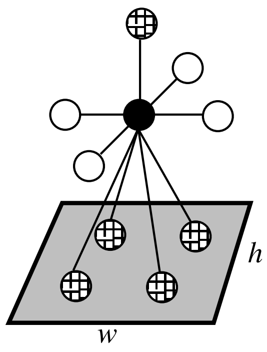

Figure 2.13: *Communication scheme of the hierarchical model*

a maximum of $(1 + 1/w)WH$ processors (cf. Equation (2.107)), since all levels must be stored at the same time. The memory (or number of processors) required for the storage of these levels (see Figure 2.12), considering a rectangular shape, is given by:

$$WH + \frac{WH}{wh} + \frac{WH}{(wh)^2} + \cdots + \frac{WH}{(wh)^M} = WH \sum_{i=0}^{M} \frac{1}{(wh)^i} < \left(1 + \frac{1}{w}\right) WH \qquad (2.107)$$

Communication Considering only the first and second order cliques (mostly used in practice, see Figure 2.13), it is clear that we have $(wh + 1)$ more communications per processor. Each site interacts with its ancestor (there is one) and its descendants (there are wh).

It turns out that the new model requires more processors and more computing time than a monogrid model. However, as we will see later in Section 2.7, experiments show that the results obtained by the hierarchical model are usually better.

2.6.4 A Hierarchical Segmentation Model

Considering the segmentation model presented in Section 2.3, its hierarchical equivalent can be derived from Equation (2.105) and Equation (2.106) [74, 75, 73]:

$$\bar{U}_1(\bar{\omega}, \mathcal{F}) = \sum_{i=0}^{M} \sum_{s^i \in \mathcal{S}^i} V_1^i(\xi^i, \mathcal{F}) \qquad (2.108)$$

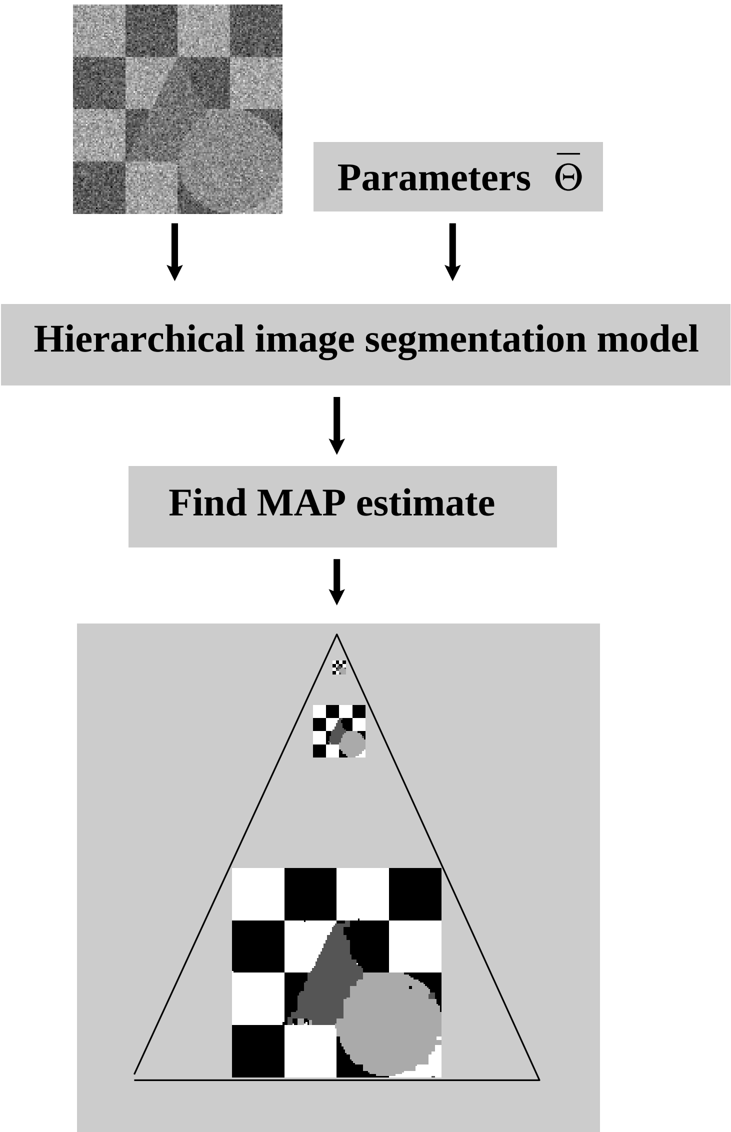

Figure 2.14: *Hierarchical supervised segmentation process.*

$$\text{and } \bar{U}_2(\bar{\omega}) = \sum_{i=0}^{M} \sum_{C^i \in \mathcal{C}^i} V_2^i(\xi_{C^i}^i) + \sum_{C \in \bar{\mathcal{C}}_3} \bar{V}_2(\bar{\omega}_c) \tag{2.109}$$

$$\text{where } \bar{V}_2(\bar{\omega}_c) = \bar{V}_{\{\bar{s},\bar{r}\}}(\bar{\omega}_{\bar{s}}, \bar{\omega}_{\bar{r}}) = \begin{cases} -\gamma & \text{if } \bar{\omega}_{\bar{s}} = \bar{\omega}_{\bar{r}} \\ +\gamma & \text{if } \bar{\omega}_{\bar{s}} \neq \bar{\omega}_{\bar{r}} \end{cases} \tag{2.110}$$

where V_1^i and V_2^i are defined in Equation (2.92) and Equation (2.93). We have a new model parameter γ which favors similar classes between a site, its ancestor and its descendants. Thus, taking into account the new parameter, Equation (2.53) can be rewritten as

$$\bar{\Theta} = \begin{pmatrix} \bar{\vartheta}_0 \\ \bar{\vartheta}_1 \\ \vdots \\ \bar{\vartheta}_{2L+1} \end{pmatrix} \equiv \begin{pmatrix} \mu_0 \\ \mu_1 \\ \vdots \\ \mu_{L-1} \\ \sigma_0 \\ \vdots \\ \sigma_{L-1} \\ \beta \\ \gamma \end{pmatrix} \tag{2.111}$$

In Figure 2.14, we give an overview of a hierarchical image segmentation process. We have two inputs: the image itself and the parameters $\bar{\Theta}$. After building the label-pyramid, they give an energy function defined in Equation (2.108)–Equation (2.110). To find the minimum of this function (that is the MAP estimate of the label-pyramid), we essentially use the same algorithms as in the monogrid case (Iterated Conditional Mode on Figure 2.14). The resulting image is the labeling at lowest level of the pyramid.

2.7 Experimental Results

We compare the Gibbs sampler [46] and Iterated Conditional Mode [13, 69] using three models for each algorithm (monogrid, multiscale and hierarchical). All tests have been conducted on a Connection Machine CM200 [65] with $8K$ processors. In the tables (see Appendix 2.B), we give for each model and for each method the number of levels in the pyramid (for the monogrid model, this is 1), the Virtual Processor Ratio (VPR) [65], the initial temperature (for the hierarchical model, this is not the same at each level, using the new MTA[6] schedule!), the number of iterations, the computing

[6]Multi-Temperature Annealing. A new annealing scheme for the energy minimization of hierarchical models. It assigns different temperatures to different levels. For more details about the algorithm, see Chapter 3.

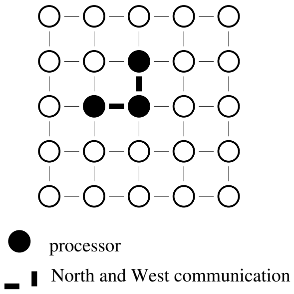

Figure 2.15: *NEWS communication scheme on the Connection Machine.*

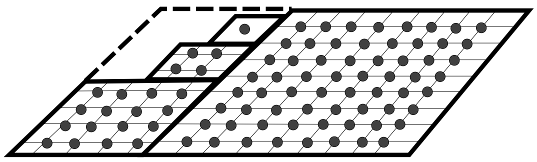

Figure 2.16: *Flat pyramid on the Connection Machine.*

time, the classification error (= the number of misclassified pixels) and the parameter β (see Equations (2.52), (2.92), (2.93)) and γ (see Equation (2.110)). In all cases, the execution has been stoped when the energy change ΔU was less than 0.1% of the current value of U.

Before comparing the models, we briefly describe the architecture of the Connection Machine and point out its limitations in the case of the hierarchical model.

2.7.1 The Connection Machine

The Connection Machine [65, 109] is a data parallel (*Single Instruction Multiple Data*—SIMD) computing system associating one processor with each pixel. This computing style is well adapted to early vision problems where a large mass of data need to be processed. On the other hand, algorithms usually require the same local computations on a small neighborhood of each pixel.

An important feature of the Connection Machine is the *virtual processor* facility. This means that a program can assume to be available any appropriate number of processors (*virtual processors*) and the machine will map it onto *physical processors*. The *Virtual Processor Ratio (VPR)* indicates how many times each physical processor must perform a task in order to simulate the appropriate number of virtual processors. Indeed, the greater the VPR, the more time consuming the computation.

As MRF models require computation over a small neighborhood of each pixel, fast interprocessor communication capability is especially important. The Connection Machine offers an efficient nearest-neighbor communication called NEWS ("North, East, West, South", see Figure 2.15). Thanks to a specialized hardware support, NEWS grids of any dimension can be handled with great speed. If we are working with monogrid or

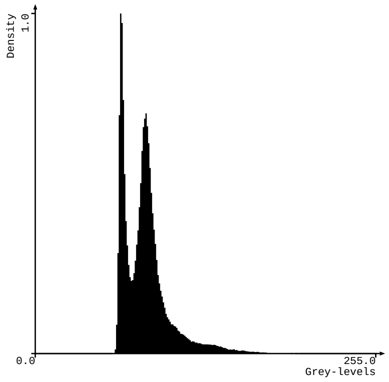

Figure 2.17: *Histogram of the "assalmer" image with 6 classes*

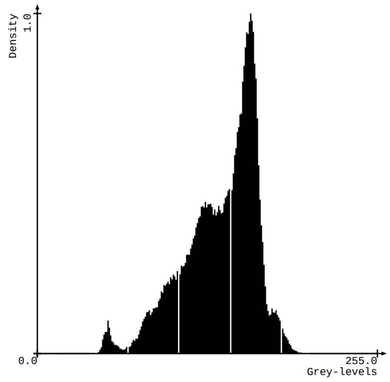

Figure 2.18: *Histogram of the "holland" image with 10 classes*

multigrid models, we use this type of communication. If we have no such regularity in the model, we have to use the general communication via the router. In this case, each processor can send data to or receive data from any other processor. Of course, the time required to deliver the message is much larger than in the previous case. For the hierarchical model, we must use this type of communications for the inter-level interactions. This is the main reason of the high computer time needed for the optimization of the associated energy function (see Appendix 2.B). The pyramid is mapped into the virtual processors as a *flat pyramid* shown in Figure 2.16. We could achieve better computer times using a *pyramid computer* (for some examples, see [71]).

2.7.2 Comparison of the Models

First, we have tested the models on noisy synthetic images of size 128×128. The first one is a checkerboard image (see Figure 2.19, Table 2.3) with 2 classes and a SNR (see Equation (1.65) for its definition) equals to $-5dB$. Note that the multiscale model gives better results than the monogrid one, especially with the ICM algorithm. This type of image is well adapted for the multiscale model, because the image has a rectangular structure similar to the model itself. The hierarchical model gives the best results with both algorithms but the computing time is much greater than for the multiscale or monogrid model. The reason is that in the hierarchical case, the whole pyramid is stored at the same time yielding a greater VPR ratio. On the other hand, we cannot use the fast "NEWS" communication scheme [65] as in the other cases.

In the second image, we have added different geometrical forms (circle and triangle) to the checkerboard image(see Figure 2.20, Table 2.4). In this case, we studied the geometrical sensitivity of the models. The Gibbs sampler gives nearly the same result in all cases. However the ICM is more sensitive to initial conditions. The multiscale model gives better result than the monogrid one but the result is not fine in the triangle and the circle. These forms have a different structure from the block structure of the model, the initialization was wrong in these regions and the ICM was not able to correct these errors. In the hierarchical case, we have a real time communication between the levels which is able to give results closed to the ones obtained with the Gibbs sampler. Of course, this model requires more computing time than the other ones.

The third image is a checkerboard image with 16 classes (see Figure 2.21 and Table 2.5). For ICM, there is a significant improvement but for the Gibbs Sampler, we observe only a slight improvement for the multiscale and hierarchical cases.

In Figures 2.22, 2.23 and 2.24, we show some real images of size 256×256: a SPOT image with 4 classes (see Figure 2.22, Table 2.6), an indoor scene with 4 classes (see Figure 2.23) and a medical image with 3 classes (see Figure 2.24).

Next, we present a SPOT image of size 512×512 (see Figure 2.25) with ground truth data (see Figure 2.26). In the following table (Table 2.1), we give the mean (μ) and the variance (σ^2) for each class (we have 6 classes).

class	1	2	3	4	5	6
μ	65.3	81.3	75.4	98.5	82.5	129.0
σ^2	6.4	12.7	14.9	16.8	9.46	183.2

Table 2.1: *Parameters of the "assalmer" image.*

As we can see, the classes 2 and 5 have nearly the same parameters, it is difficult to distinguish between them. Figure 2.17 shows the histogram of the original image. We can clearly distinguish three peaks (at about 64, 80, and 120) but the other classes are quite mixed. Figure 2.27 (resp. Figure 2.28) shows the results obtained with the ICM (resp. the Gibbs Sampler). For these results, we give a map drawn by an expert (ground truth data, see Figure 2.26). The classes $1-6$ correspond to the regions $B_{3c}, B_{3b}, B_{3d}, a_2, hc$ and 92_a on the map. For the hierarchical model a slight improvement can be noticed for the results of the Gibbs sampler, however, for the ICM, the improvement is more significant. In Table 2.7 we give the parameters and the computing time for each model and each method.

Finally, we present another SPOT image with 10 classes (Figure 2.29 – Figure 2.32,

Table 2.8). The ground truth data is showed on each images[7]. In the following table (Table 2.2), we give the mean (μ) and the variance (σ^2) for each class. Figure 2.18 shows the histogram of the original image. As for the previous image, the results are slightly better for the hierarchical model.

class	1	2	3	4	5	6	7	8	9	10
μ	54.61	73.57	159.96	122.84	129.90	146.65	82.56	100.57	93.85	182.34
σ^2	93.10	4.10	31.31	8.90	37.42	15.83	35.58	308.86	93.71	73.18

Table 2.2: *Parameters of the "holland" image.*

[7]The regions are drawn by an expert. Unfortunately, they are shifted up by some pixels. Please take it into account when evaluating the results.

Appendix

2.A Images

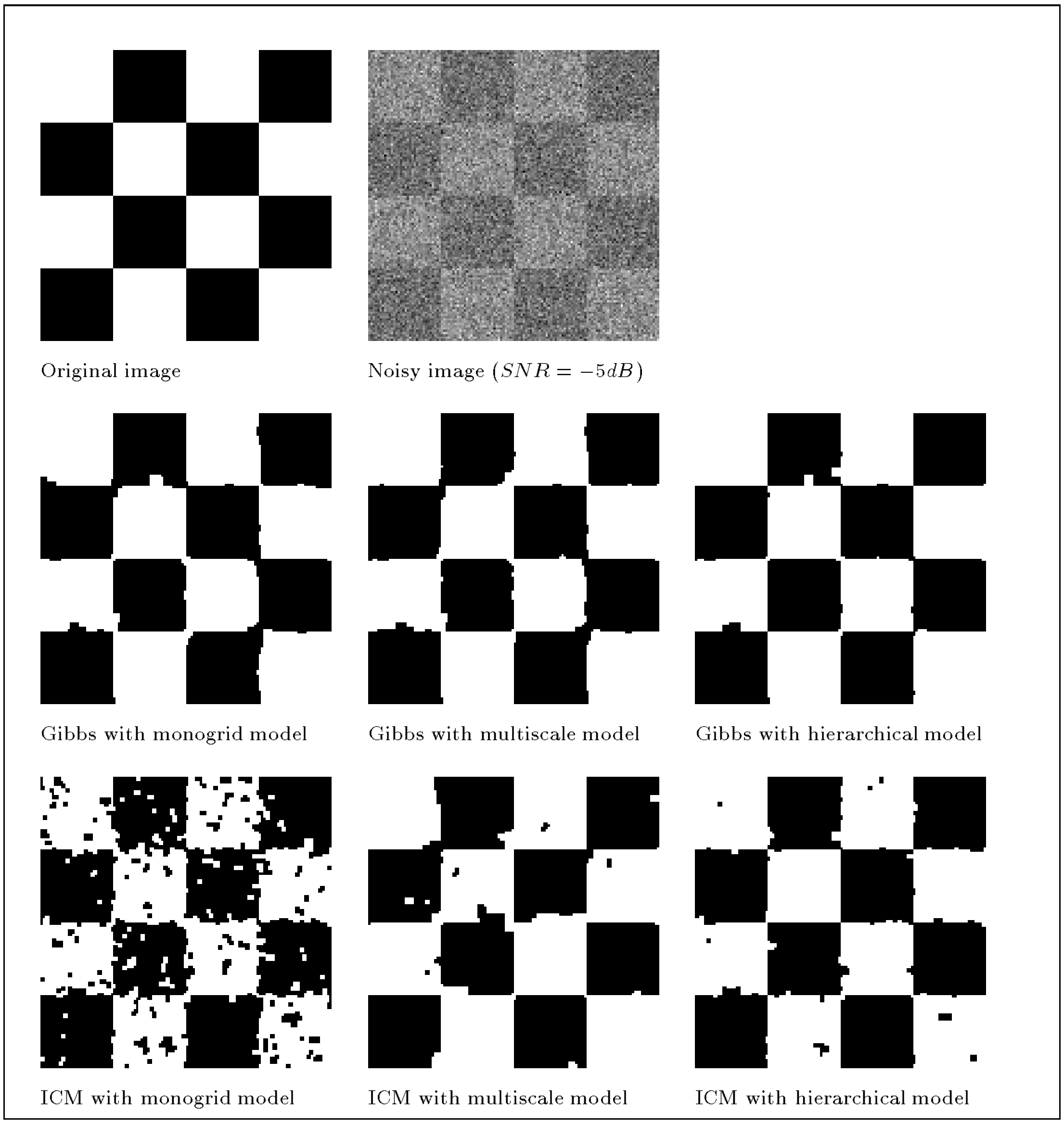

Figure 2.19: *Results on the "checkerboard" image with 2 classes.*

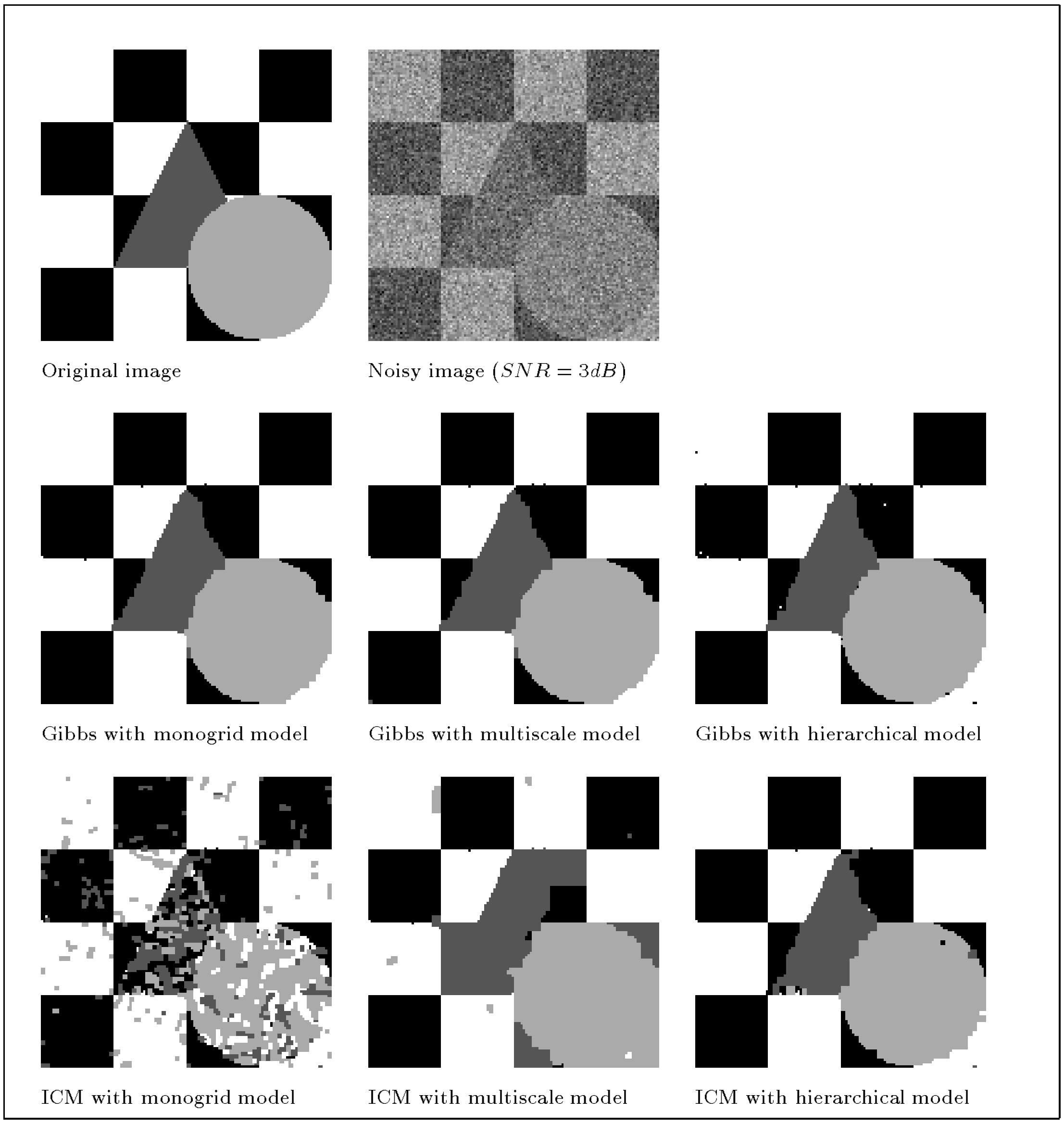

Figure 2.20: *Results on the "triangle" image with 4 classes.*

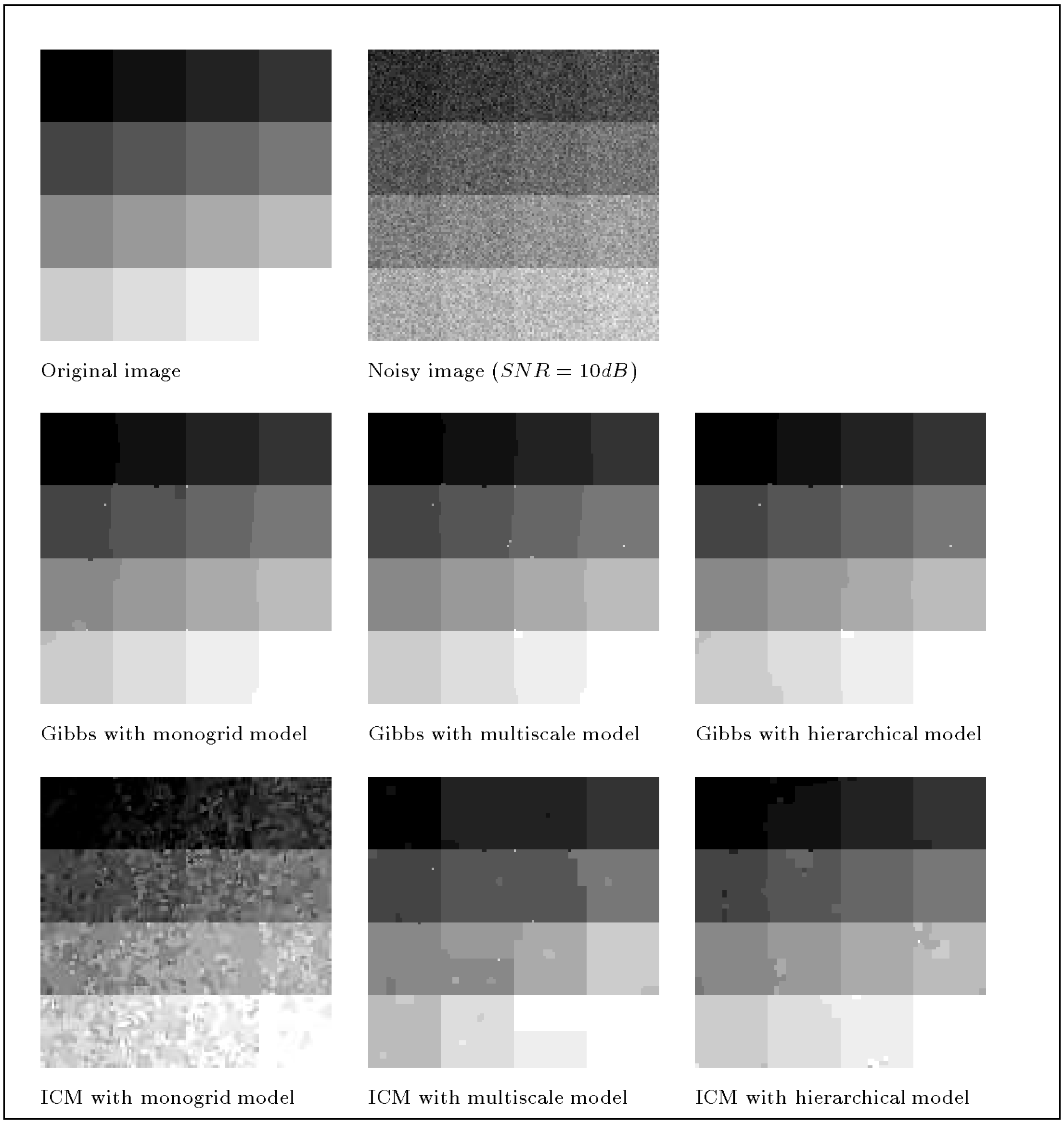

Figure 2.21: *Results on the "grey-scale" image with 16 classes.*

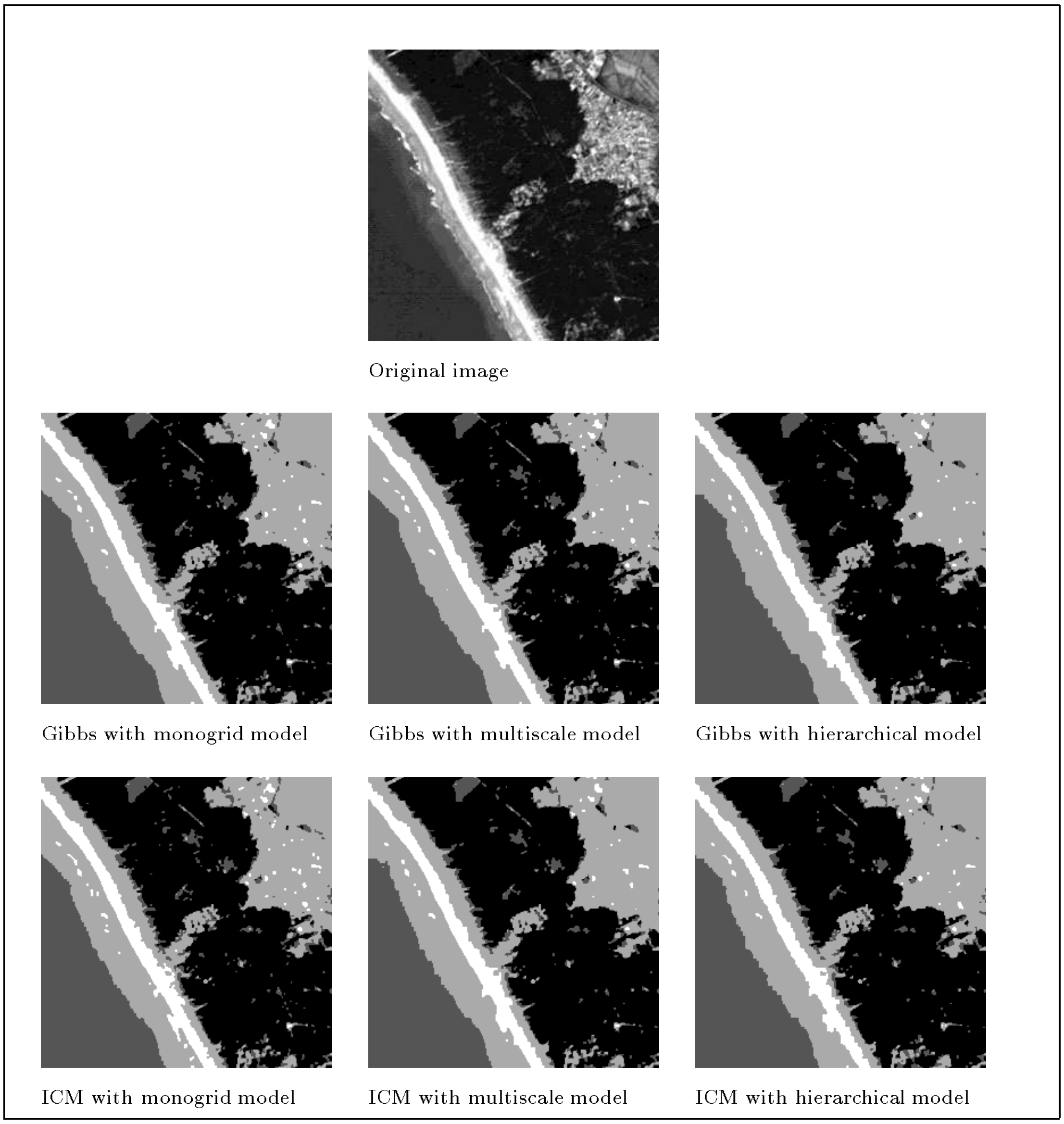

Figure 2.22: *Results on the "SPOT" image with 4 classes.*

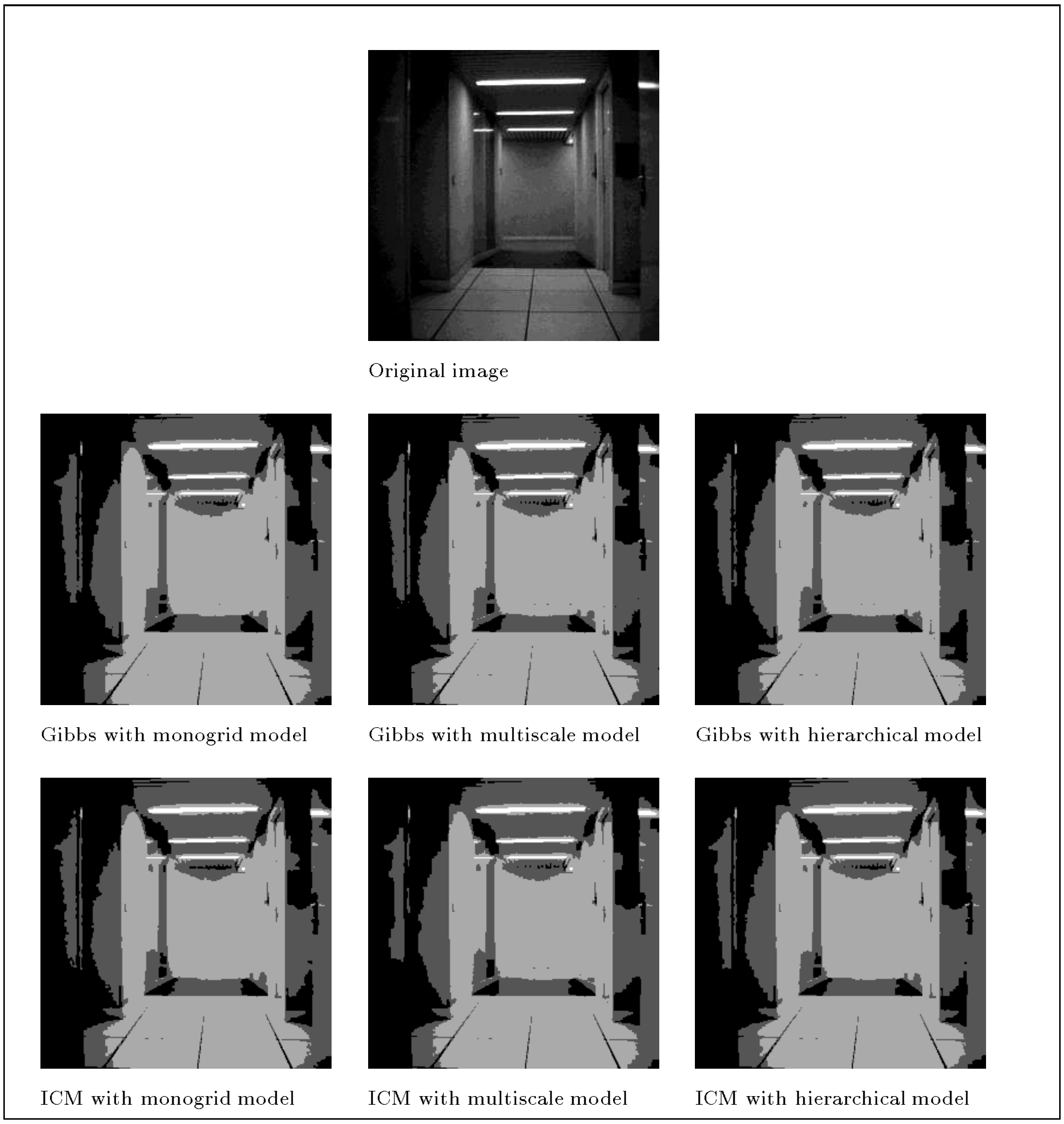

Figure 2.23: *Results on an indoor scene ("couloir") with 4 classes.*

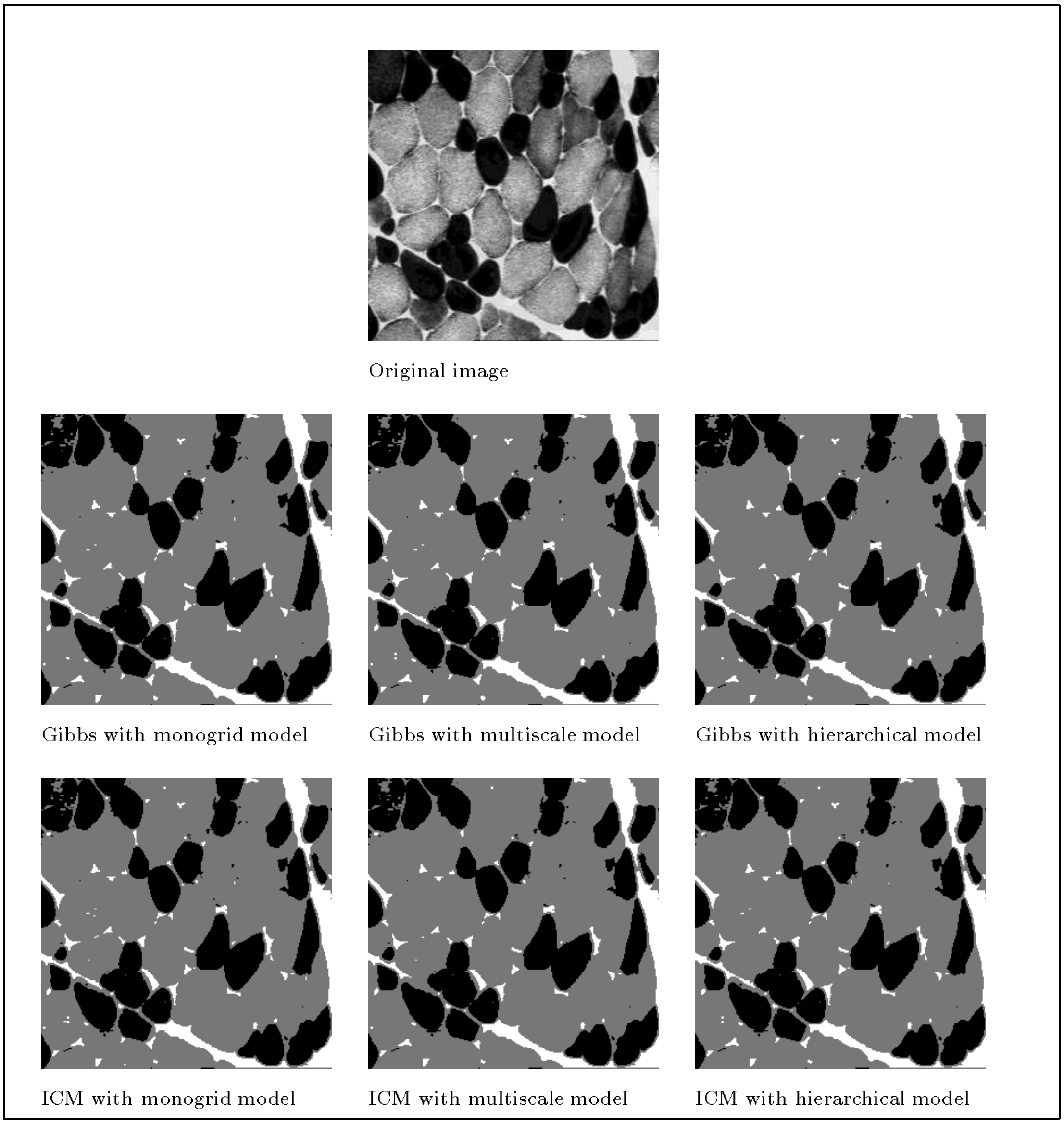

Figure 2.24: *Results on a medical image ("muscle") with 3 classes.*

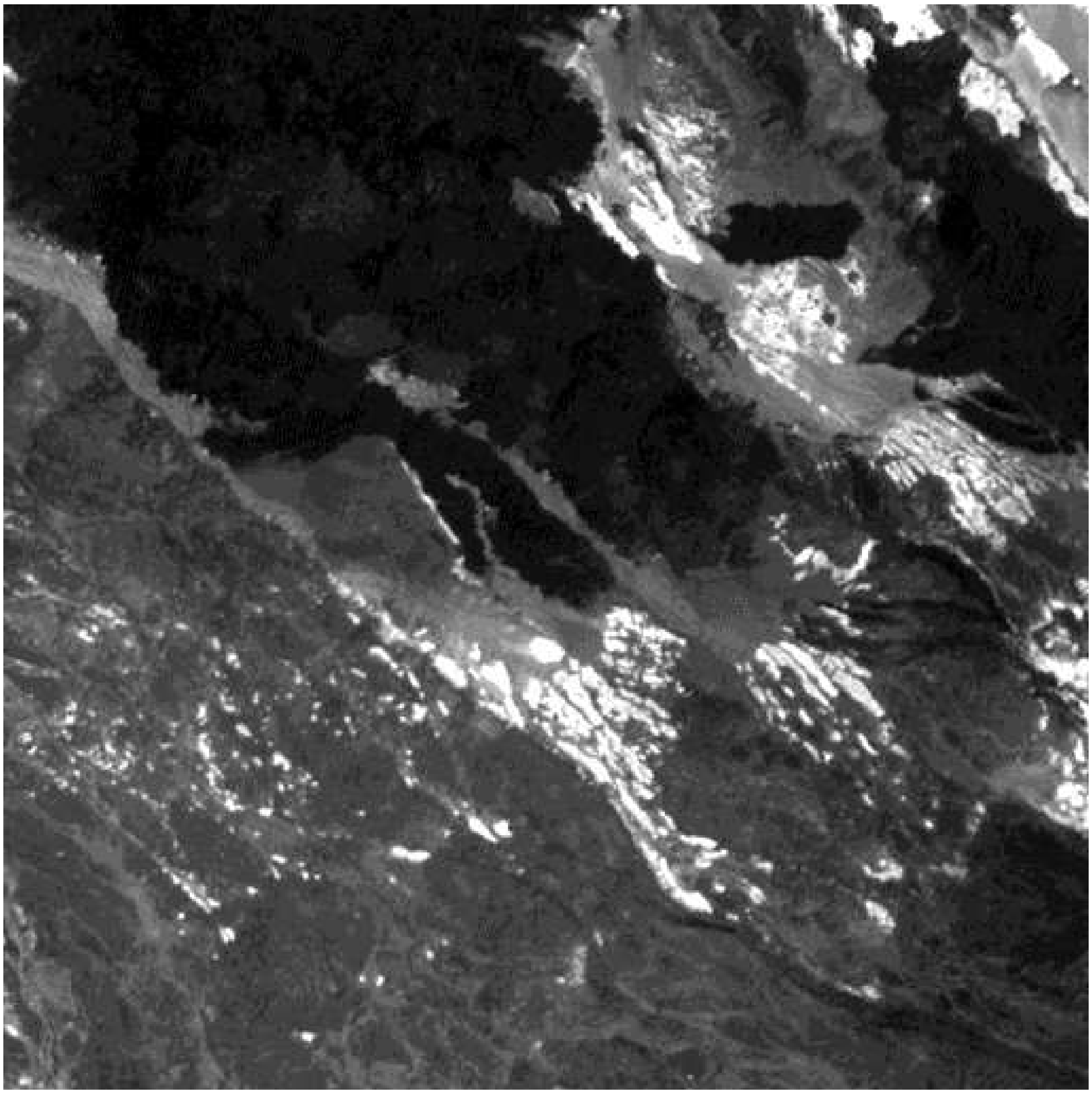

Figure 2.25: *Original SPOT image "assalmer" with 6 classes.*

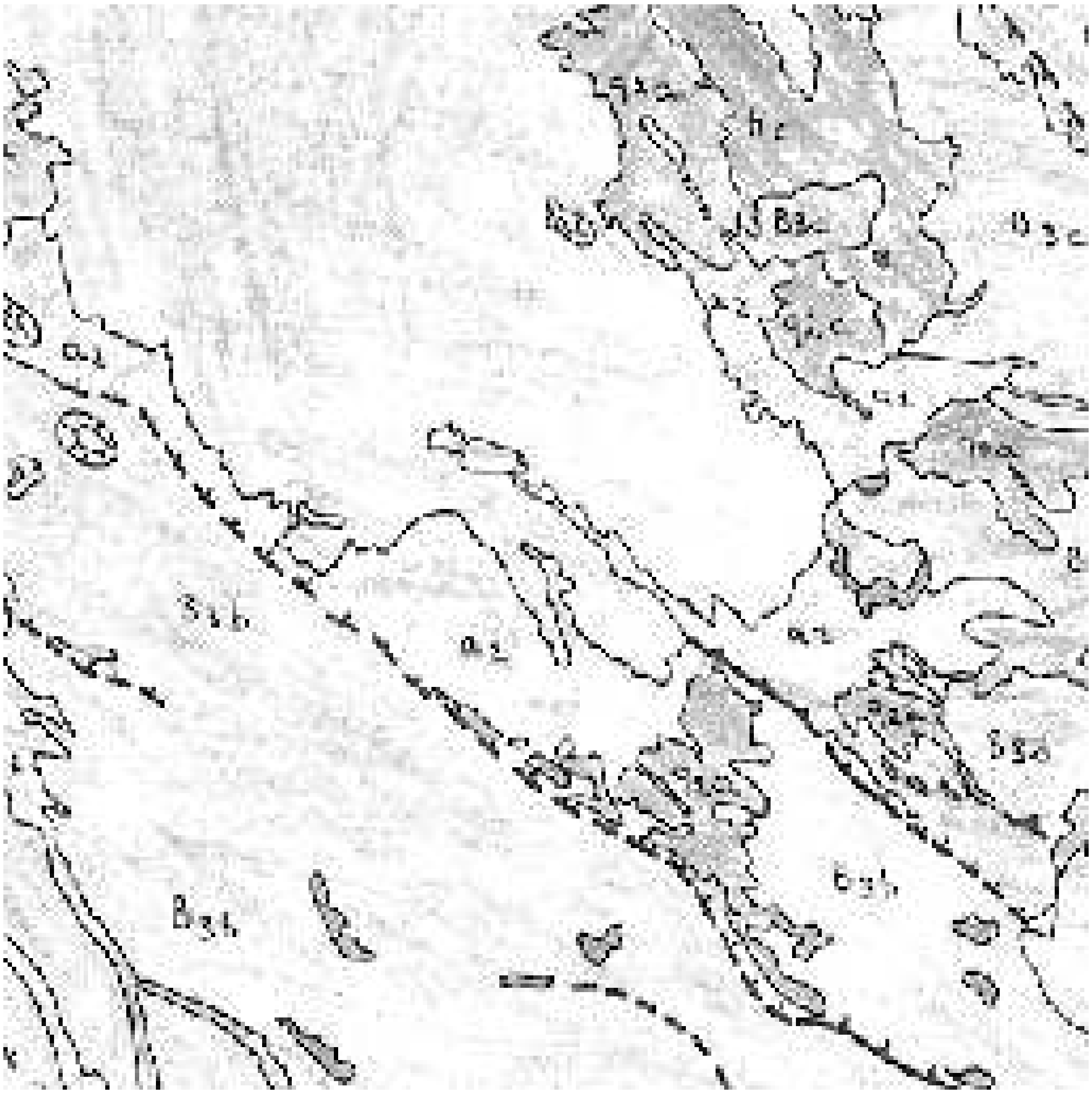

Figure 2.26: *Ground truth data.*

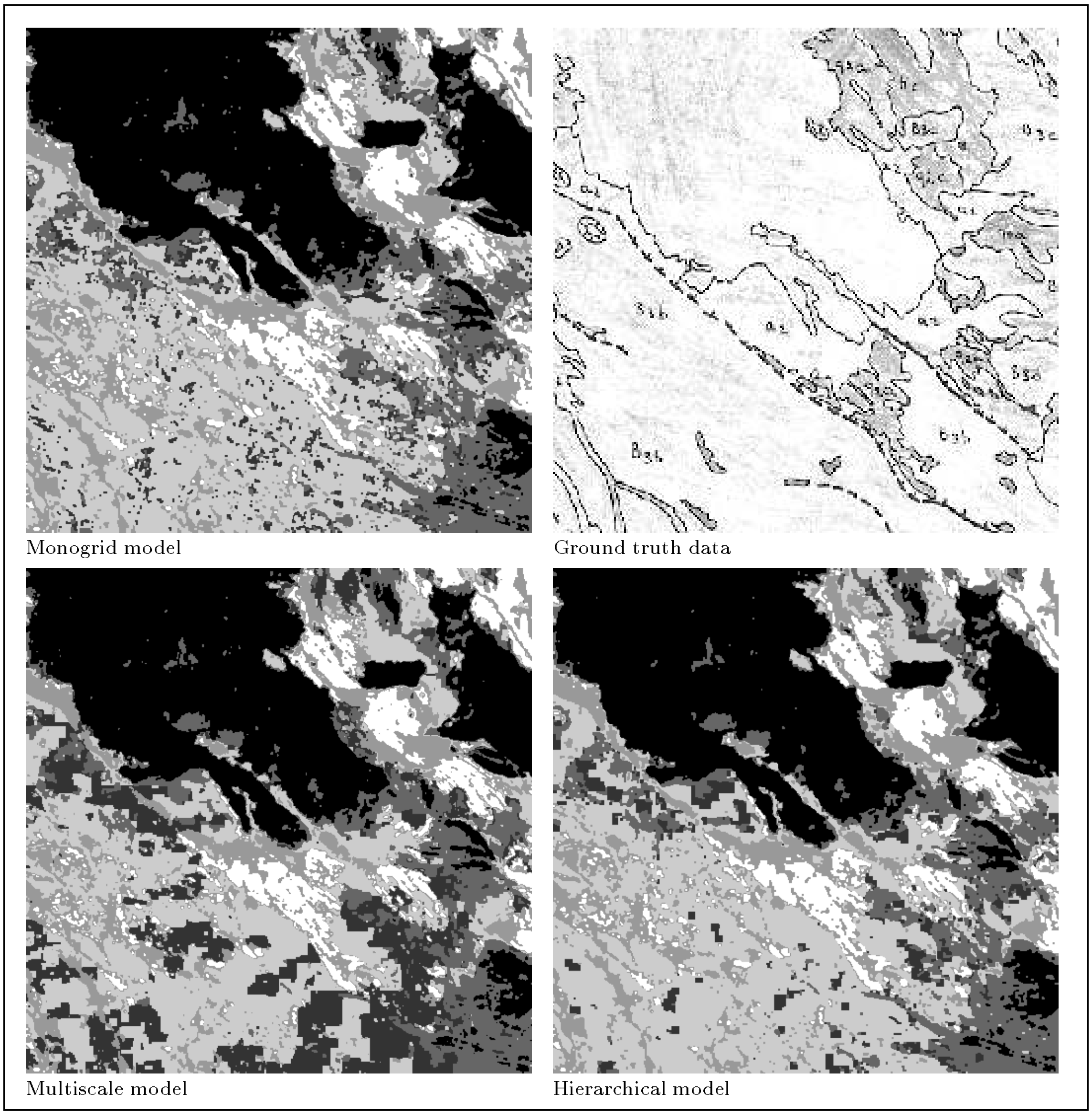

Figure 2.27: *Results of the ICM algorithm. Comparison with ground truth data.*

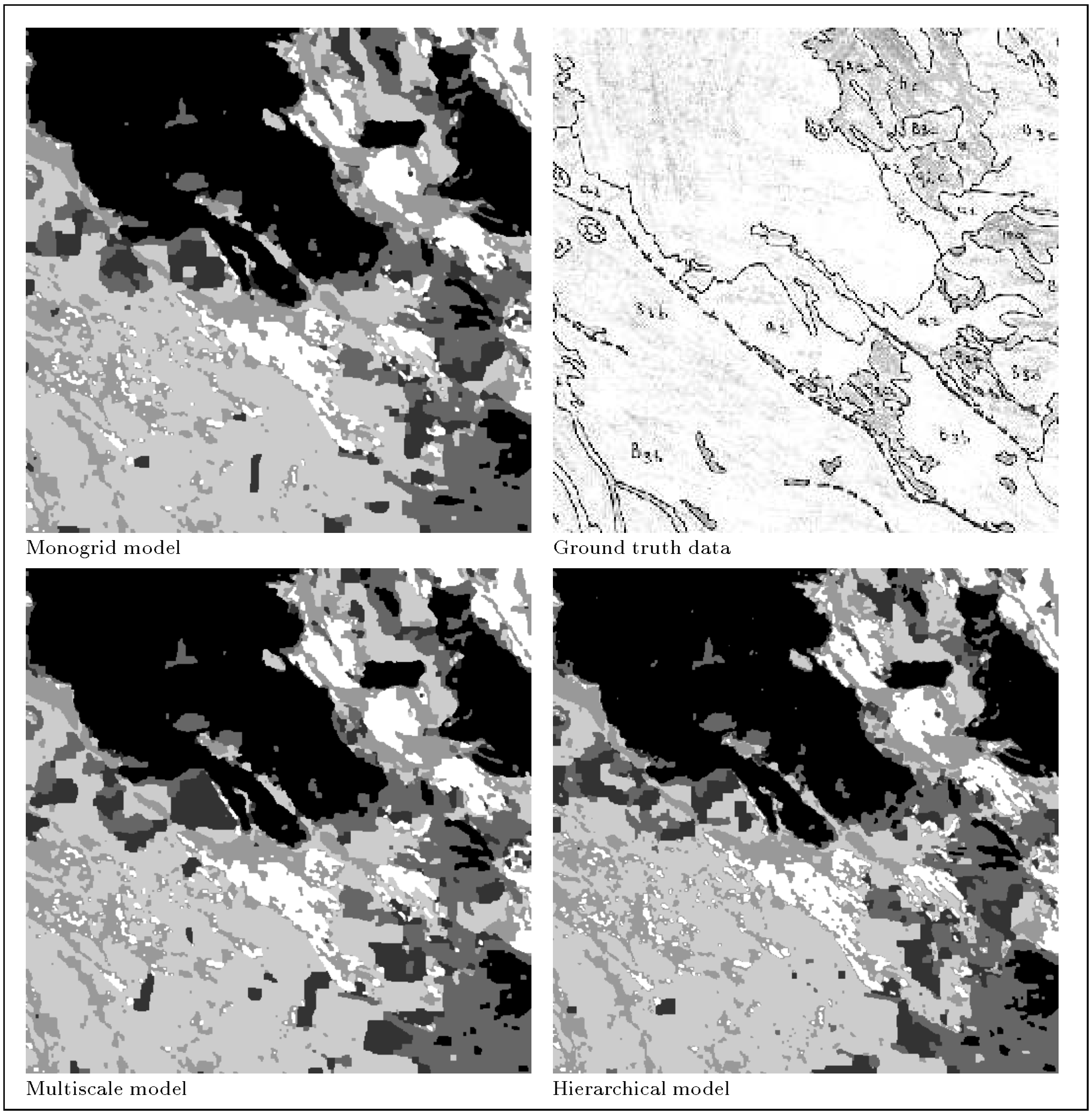

Figure 2.28: *Results of the Gibbs Sampler. Comparison with ground truth data.*

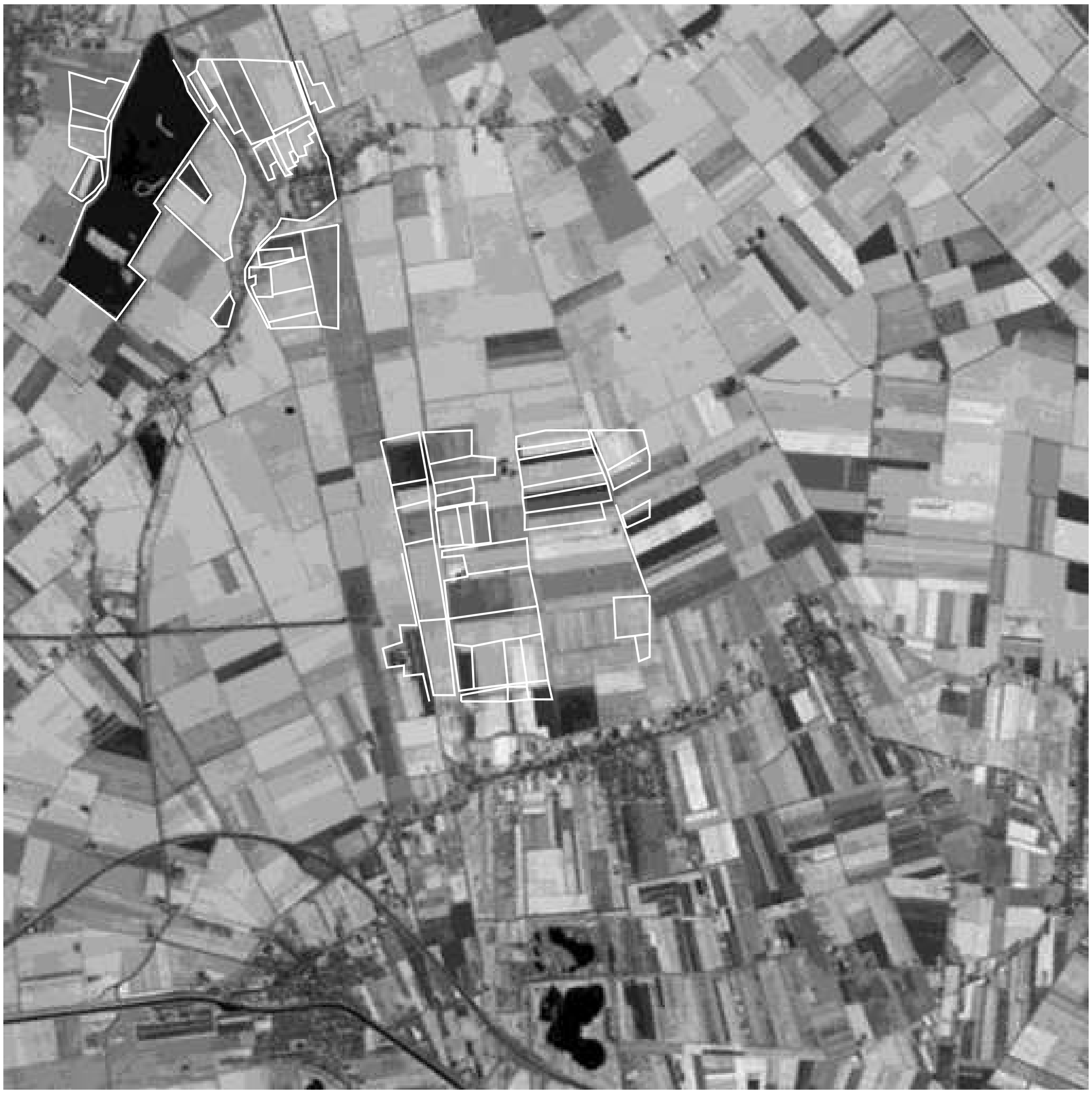

Figure 2.29: *Original SPOT image "holland".*

Figure 2.30: *Monogrid segmentation result with* 10 *classes (ICM).*

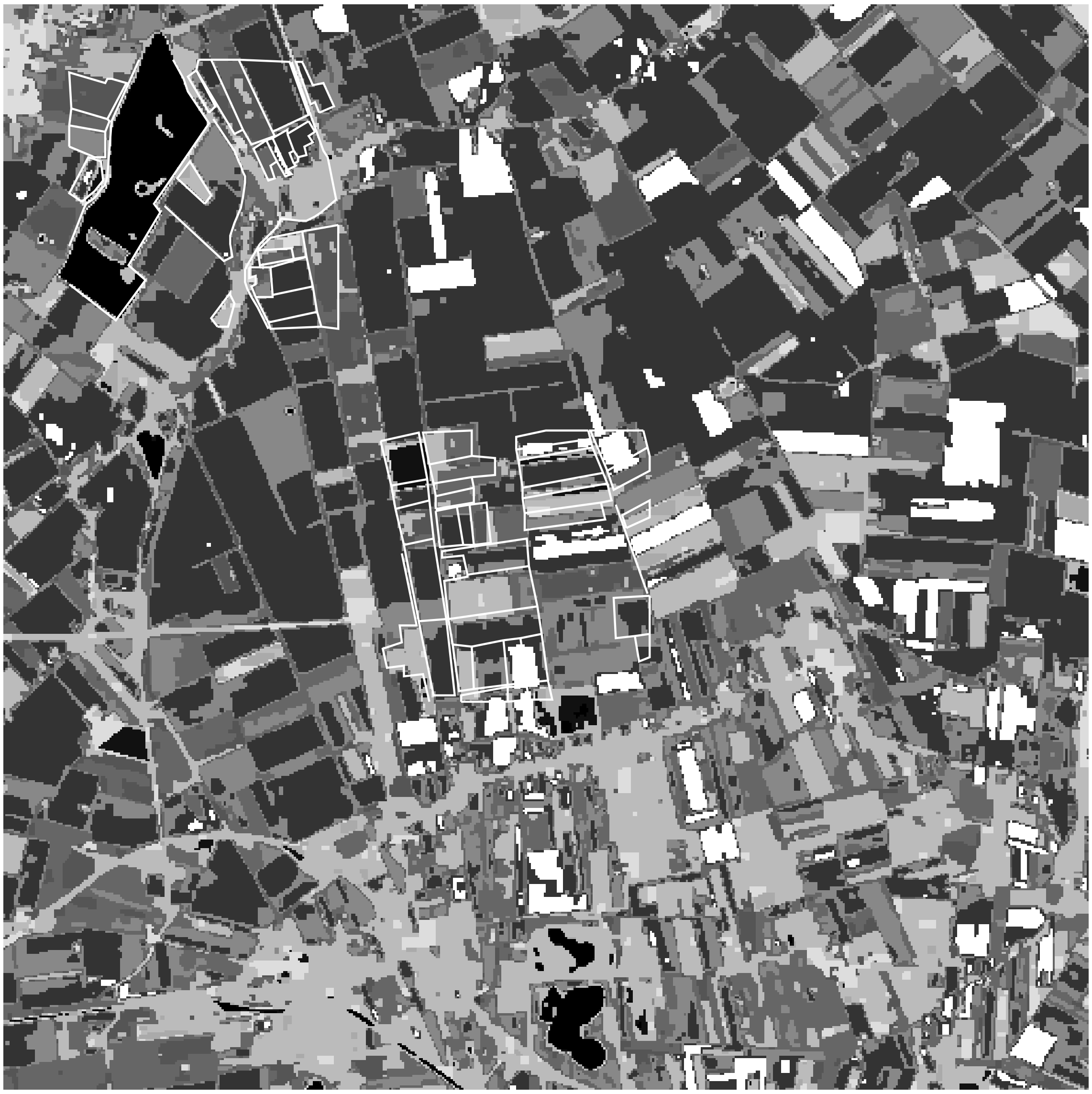

Figure 2.31: *Multiscale segmentation result with* 10 *classes (ICM).*

Figure 2.32: *Hierarchical segmentation result with* 10 *classes (ICM).*

2.B Tables

monogrid	levels	VPR	T_0	iter.	total time	time/iter.	error	β	γ
Gibbs	1	2	4	62	1.91 sec.	0.03 sec.	260 (1.59%)	0.9	—
ICM	1	2	1	8	0.077 sec.	0.009 sec.	1547 (9.44%)	0.9	—
multiscale	—	—	—	—	—	—	—	—	—
Gibbs	4	1,2	4	136	3.25sec.	0.02 sec.	236 (1.44%)	0.7	—
ICM	4	1,2	1	18	0.14 sec.	0.008 sec.	465 (2.83%)	0.7	—
hierarchical	—	—	—	—	—	—	—	—	—
Gibbs	4	4	4,3,2,1	23	50.1 sec.	2.18 sec.	115 (0.7%)	0.7	0.3
ICM	4	4	1	11	16.6 sec.	1.5 sec.	300 (1.83%)	0.7	0.3

Table 2.3: *Results on the "checkerboard" image* (128×128) *with 2 classes.*

monogrid	levels	VPR	T_0	iter.	total time	time/iter.	error	β	γ
Gibbs	1	2	4	68	3.01 sec.	0.04 sec.	183 (1.12%)	1.0	—
ICM	1	2	1	9	0.15 sec.	0.02 sec.	2948 (17.99%)	1.0	—
multiscale	—	—	—	—	—	—	—	—	—
Gibbs	4	1,2	4	101	3.85sec.	0.04 sec.	176 (1.07%)	1.0	—
ICM	4	1,2	1	17	0.22 sec.	0.01 sec.	1657 (10.11%)	0.9	—
hierarchical	—	—	—	—	—	—	—	—	—
Gibbs	4	4	4,3,2,1	41	141.97 sec.	3.46 sec.	191 (1.16%)	0.7	0.1
ICM	4	4	1	11	30.17 sec.	2.74 sec.	293 (1.78%)	0.8	0.5

Table 2.4: *Results on the "triangle" image* (128×128) *with 4 classes.*

monogrid	levels	VPR	T_0	iter.	total time	time/iter.	error	β	γ
Gibbs	1	2	4	201	22.96 sec.	0.12 sec.	340 (2.08%)	1.0	—
ICM	1	2	1	10	0.55 sec.	0.05 sec.	8721 (53.22%)	1.0	—
multiscale	—	—	—	—	—	—	—	—	—
Gibbs	4	1,2	4	337	32.97sec.	0.1 sec.	331 (2.02%)	1.0	—
ICM	4	1,2	1	15	0.68 sec.	0.05 sec.	5198 (31.73%)	1.0	—
hierarchical	—	—	—	—	—	—	—	—	—
Gibbs	4	4	4,3,2,1	107	1169.46 sec.	10.93 sec.	316 (1.93%)	1.0	0.2
ICM	4	4	1	16	162.87 sec.	10.18 sec.	795 (4.85%)	1.0	0.5

Table 2.5: *Results on the "grey-scale" image* (128×128) *with* 16 *classes.*

monogrid	levels	VPR	T_0	iter.	total time	time/iter.	β	γ
Gibbs	1	8	4	64	9.06 sec.	0.14 sec.	2.0	—
ICM	1	8	1	7	0.33 sec.	0.047 sec.	2.0	—
multiscale	—	—	—	—	—	—	—	—
Gibbs	4	1-8	4	106	10.22 sec.	0.09 sec.	2.0	—
ICM	4	1-8	1	37	1.14 sec.	0.03 sec.	1.0	—
hierarchical	—	—	—	—	—	—	—	—
Gibbs	4	16	4,3,2,1	29	353.54 sec.	12.19 sec.	2.0	0.4
ICM	4	16	1	6	58.59 sec.	9.76 sec.	0.5	0.6

Table 2.6: *Results on the "SPOT" image* (256×256) *with* 4 *classes.*

monogrid	levels	VPR	T_0	iter.	total time	time/iter.	β	γ
Gibbs	1	32	4	234	163.18 sec.	0.69 sec.	1.5	—
ICM	1	32	1	8	2.03 sec.	0.25 sec.	1.5	—
multiscale	—	—	—	—	—	—	—	—
Gibbs	5	1-32	4	580	180.17 sec.	0.31 sec.	1.5	—
ICM	5	1-32	1	36	5.15 sec.	0.14 sec.	0.3	—
hierarchical	—	—	—	—	—	—	—	—
Gibbs	5	64	4,3,2,1	154	9629.33 sec.	62.53 sec.	0.7	0.1
ICM	5	64	1	16	915.99 sec.	57.25 sec.	1.0	0.2

Table 2.7: *Results on the "assalmer" image* (512×512) *with* 6 *classes.*

ICM	levels	VPR	iter.	total time	time/iter.	β	γ
monogrid	1	32	9	3.56 sec.	0.39 sec.	0.5	—
multiscale	5	1-32	47	10.14 sec.	0.22 sec.	0.5	—
hierarchical	5	64	21	1900.83 sec.	90.52 sec.	0.8	0.4

Table 2.8: *Results on the "holland" image* (512×512) *with* 10 *classes.*

In This Chapter:

3. Optimization

Bayesian methods coupled with Markovian modelization usually result in a non-convex cost (or energy) function. In order to find an estimate, one has to optimize this function. Unfortunately, this is a very hard computational problem known as *combinatorial optimization*. For example, considering an image 16×16 with only two possible labels at each pixel, we obtain a configuration space of 2^{256} elements. It is then impossible to find the optimum by computing the possible values of the cost function. On the other hand, due to the non-convexity classical gradient descent methods cannot be used since they get stuck in a local minimum.

The idea of the solution comes again from the statistical physics: In 1953, Metropolis *et al.* [93] proposed a Monte-Carlo simulation to find equilibrium states of thermodynamical systems. It was realized in the early 80's, independently by Černy [24] and Kirkpatrick *et al.* [82], that there is an analogy between minimizing the cost function of a combinatorial optimization problem and finding energy minima of thermodynamical systems by slowly cooling a solid until equilibrium is reached. They have substituted the energy function of the solid by the cost function and executed the Metropolis algorithm at a sequence of slowly decreasing "temperature". The so defined combinatorial optimization algorithm was named *Simulated Annealing* (SA) [83, 104, 44].

The research in the field has rapidly grown up resulting in a variety of contributions to the original SA. The most important is probably the *Gibbs Sampler* proposed by Geman and Geman in [46]. While SA algorithms find the global optimum, they require a large amount of computation. To avoid this drawback, two solutions have been proposed: One of them deals with the possible parallelization of SA algorithms [5]. Another solution is to use *deterministic* algorithms which are suboptimal but converge in a few iterations requiring less computing time [13, 79].

3.1 Equilibrium State and the Metropolis Algorithm

For historical reasons, we begin this chapter by the original formulation of the Metropolis algorithm, as it was proposed by Metropolis *et al.* Following [93], we are given N particles on a square. If the positions of the particles are known, we can easily calculate the potential energy of the system:

$$E = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1, j\neq i}^{N} V(\delta_{ij}), \tag{3.1}$$

where V is the potential between molecules and δ_{ij} is the distance between particles i and j. To calculate the equilibrium value of any quantity F, the following integral has to be computed [93]:

$$\bar{F} = \frac{\int F \exp\left(\frac{-E}{kT}\right) d^{2N}p d^{2N}q}{\int \exp\left(\frac{-E}{kT}\right) d^{2N}p d^{2N}q} \tag{3.2}$$

where $d^{2N}pd^{2N}q$ is a volume element in the $4N$-dimensional phase space.

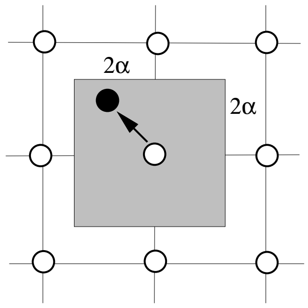

Figure 3.1: *Moving particles according to the Metropolis algorithm.*

To compute this integral, the following procedure has been suggested: Place the N particles in any configuration in a lattice where each particle is represented by its coordinates (x, y). According to the following rules, we move each particles within a square of side 2α centered about its original position (see Figure 3.1):

$$x^{new} = x + \alpha\xi_1 \tag{3.3}$$

$$y^{new} = y + \alpha\xi_2 \tag{3.4}$$

where ξ_1 and ξ_2 are uniform random numbers in the range $[-1, 1]$. Then, we compute the energy-change ΔE of the system caused by the move. If the move brings the system to a state of lower energy (i.e. if $\Delta E < 0$), then the particle is placed in its new position. If $\Delta E > 0$, we accept the new position with probability $\exp(-\Delta E/kT)$: Let ξ_3 be a random number in the range $[0, 1]$. If $\xi_3 < \exp(-\Delta E/kT)$, we move the particle to (x^{new}, y^{new}). Otherwise, we keep its old position. The integral in Equation (3.2) is then approximated by

$$\bar{F} = \frac{1}{M}\sum_{i=1}^{N} F_i, \tag{3.5}$$

where F_i is the property F of the system after the i^{th} move has been carried out. It has been proved in [93] that the above procedure generates configurations with probability $\exp(-E/kT)$.

3.2 Combinatorial Optimization and Simulated Annealing

The classic example of a combinatorial optimization problem is the *traveling salesman problem*:

Example 3.2.1 Given N cities and the distances D_{ij} between the cities i and j representing the cost of traveling. One has to plan the sales-man's *optimal* route which will pass through each city once and return finally to the starting point, minimizing the total length.

The above example belongs to the class of NP-complete problems. As it is well known, there is no method to find an exact solution of such a problem with a computing effort bounded by a power of N. SA is one of the heuristics proposed to solve the traveling salesman problem.

As we have said in the introduction, SA algorithm is based on the analogy between the simulation of the annealing of solids and the solving of combinatorial optimization problems. This is why the algorithm proposed by Černy [24] and Kirkpatrick *et al.* [82] became known as *Simulated Annealing*.

In physics, annealing means heating up a solid to a maximum value at which all particles randomly arrange themselves in the liquid phase then slowly cooling it down. In this way, all particles arrange themselves in the low energy state of a corresponding lattice. At each temperature T, the solid is allowed to reach a *thermal equilibrium* which is characterized by the *Boltzmann distribution*:

$$P(\omega) = \frac{1}{Z(T)} \exp\left(-\frac{U(\omega)}{kT}\right), \tag{3.6}$$

where $U(\omega)$ is the energy of the state, $Z(T)$ is the partition function depending on T and k is the Boltzmann constant. For a fixed T, the above equation is nothing else but a Gibbs distribution. Clearly, as the temperature decreases, the above distribution concentrates on the states with lower energy and when the temperature approaches zero, only the minimum states have a non-zero probability. Decreasing the temperature is crucial: It has been pointed out in [82] that if the cooling is too rapid and the system is not allowed to reach thermal equilibrium for each temperature, a global minimum cannot be reached.

For a fixed temperature, the evolution to thermal equilibrium is simulated by the *Metropolis algorithm* [93] presented in Section 3.1. Here, the algorithm is used to generate sequences of configurations of a combinatorial optimization problem. SA is then a sequence of *Metropolis algorithms* evaluated at a sequence of *decreasing* temperatures such that equilibrium is reached at each temperature[1].

Now, it is time to give an exact definition of the SA algorithm in its original formulation: Let us denote by $\omega, \eta, \ldots$ the configurations of a combinatorial optimization problem (they correspond to the states of a solid) and let $U(\omega)$ denotes the cost (also called energy) of the configuration ω (it corresponds to the energy of the state ω in a thermodynamical system.). The elements of the configurations are indexed by $\mathcal{S} = \{s_1, s_2, \ldots, s_N\}$ and the common state space is denoted by $\Lambda = \{0, 1, \ldots, L-1\}$. The set of all possible configurations is denoted by Ω. Since $\forall s \in \mathcal{S} : \omega_s \in \Lambda$, $\Omega = \Lambda^N$.

Algorithm 3.2.1 (Simulated Annealing)

① *Set $k = 0$ and initialize ω randomly. Choose a sufficiently high initial temperature $T = T_0$.*

② *Construct a trial perturbation η from the current configuration ω such that η differs only in one element from ω.*

③ **(Metropolis criteria)** *Compute $\Delta U = U(\eta) - U(\omega)$ and accept η if $\Delta U < 0$ else accept with probability $\exp(-\Delta U/T)$ (analogy with thermodynamics):*

$$\omega = \begin{cases} \eta & \text{if } \Delta U \leq 0, \\ \eta & \text{if } \Delta U > 0 \text{ and } \xi < \exp(-\Delta U/T), \\ \omega & \text{otherwise} \end{cases} \tag{3.7}$$

where ξ is a uniform random number in $[0, 1)$.

④ *Goto Step ② until equilibrium is reached.*

⑤ *Decrease the temperature: $T = T_{k+1}$ and goto Step ② with $k = k + 1$ until the system is* frozen.

[1] This is the original formulation of the SA which is practically not used nowadays. We will discuss later more recent SA variants.

The above algorithm is also called *homogeneous annealing* since it is described by a sequence of homogeneous Markov chains. If the temperature is decreased after each transition, the algorithm is described by an inhomogeneous Markov chain thus it is referred to as *inhomogeneous annealing*. This is the most often used form of annealing. We obtain such an algorithm if we withdraw Step ④ from Algorithm 3.2.1.

3.2.1 Mathematical Model

The mathematical model of the Simulated Annealing was extensively studied by Aarts and van Laarhoven in [83]. We briefly describe this model:

The SA generates a sequence of configurations which constitutes a Markov chain. $P_{\omega,\eta}(k-1,k)$ is the probability that the configuration obtained after k transitions is η given the previous configuration ω. Furthermore, let $X(k)$ denote the state reached after the k^{th} transition. The probability of this event is given by:

$$P(X(k)=\omega)=\sum_{\zeta} P(X(k-1)=\zeta)P_{\zeta,\omega}(k-1,k) \quad k=1,2,\ldots \tag{3.8}$$

If the transition probability $P_{\omega,\eta}(k-1,k)$ does not depend on k, the corresponding chain is homogeneous, otherwise it is inhomogeneous. The transition probabilities depend also on the temperature parameter T. Thus, if T is kept constant, the chain will be homogeneous and the transition matrix $P=P(T)$ can be written as:

$$P_{\omega,\eta}(T)=\begin{cases} G_{\omega,\eta}(T)A_{\omega,\eta}(T) & \forall\eta\neq\omega \\ 1-\sum_{\zeta} G_{\omega,\zeta}(T)A_{\omega,\zeta}(T) & \eta=\omega \end{cases} \tag{3.9}$$

Where $G_{\omega,\eta}(T)$ is the *generation probability* of generating η from ω and $A_{\omega,\eta}(T)$ is the *acceptance probability* of configuration η, once it has been generated from ω. It is clear from the definition in Equation (3.9) that $P(T)$ is a stochastic matrix:

$$\forall\omega:\sum_{\zeta} P_{\omega,\zeta}(T)=1 \tag{3.10}$$

In the original formulation of the algorithm, $G_{\omega,\eta}(T)$ is given by the uniform distribution on the configurations η which differs from ω only for one component. $A(T)$ is given by the Metropolis criterion:

$$A_{\omega,\eta}(T)=\min(1,\exp(-(U(\eta)-U(\omega))/T)) \tag{3.11}$$

where $U(\omega)$ is the cost or energy function.

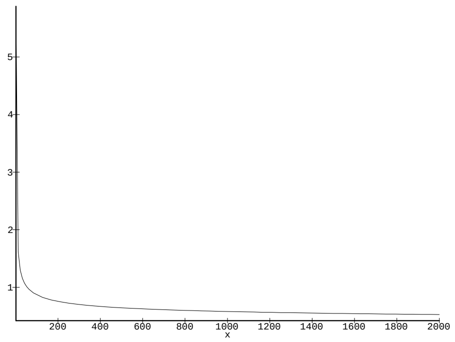

Figure 3.2: *Logarithmic cooling schedule* $(4/\ln(k))$.

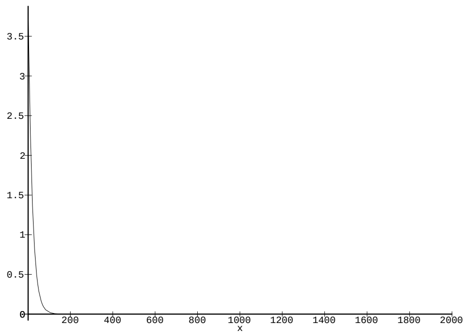

Figure 3.3: *Exponential cooling schedule* $(0.95^k \cdot 4)$.

3.2.1.1 More on Cooling Schedules

In Section 3.3, it will be shown that SA converges with probability one to a globally optimal configuration if certain conditions hold for the temperature schedule. Namely, that T_k goes towards 0 not faster than $C/\ln(k)$ for some constant C independent of k (inhomogeneous annealing). For the homogeneous annealing, there are other conditions, but we will focus on the *inhomogeneous annealing* since it is the most commonly used schedule.

It is clear that the above theoretical schedule cannot be implemented since it is too slow. Thus, we have to approximate it. Due to this approximation, the algorithm is no longer guaranteed to find a global optimum.

Initial Temperature The initial temperature T_0 must be so high that virtually all transitions are accepted. It is extremely difficult to determine such a value since it is related to the maximum and minimum values of the energy function to be minimized [46]. There are some heuristics to get a reasonably estimate of the initial temperature [82, 83] but usually one set T_0 to a relatively low value resulting in a faster execution of the algorithm. In [46] for example, $T_0 = 4$ has been suggested, and we have used the same value throughout the simulations presented in this book.

Final Temperature Obviously, $lim_{k\to\infty} T_k = 0$ can only be approximated in a finite number of values for T_k. Thus, we need a stoping criteria determining the final value of

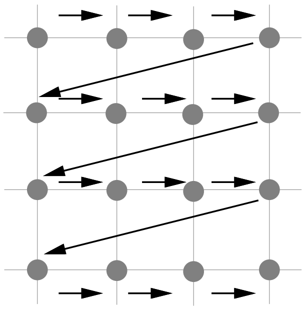

Figure 3.4: *Example of a raster scan generation mechanism.*

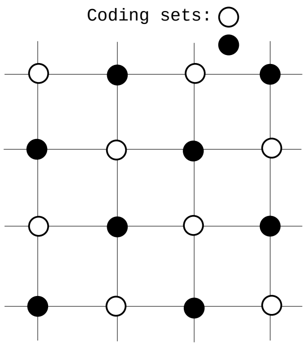

Figure 3.5: *Coding sets in the case of a first order MRF.*

the temperature. We can simply fix the number of values T_k or terminate the execution of the algorithm if the last few configurations obtained by SA have nearly the same energy (i.e. ΔU is less than a certain threshold).

Cooling Schedule The most important point is a decreasing rule of the temperature. Logarithmic rules (cf. Figure 3.2) are usually too slow for practical use. Instead, exponential rules (cf. Figure 3.3) are the most frequently used:

$$T_{k+1} = c \cdot T_k, \qquad k = 0, 1, 2, \dots \tag{3.12}$$

where $c < 1$ is a constant close to 1. This rule was first proposed in [82] with $c = 0.95$ and it became widely used by others. We have also used this rule in our experiments.

3.2.1.2 More on Generation Matrixes

In some cases, there may be better ways of generating configurations than the uniform distribution. Hereafter, we discuss some improvements in the generation mechanism.

In image processing, the most convenient implementation is a raster scan where the pixels of the image are visited for update in order $(0,0), (0,1), \dots, (1,0), (1,1) \dots$ (see Figure 3.4). At each site, a new state is chosen with a uniform probability among the possible states different from the current one.

To speed up the algorithm, one can use synchronous update (that is updating each pixel at the same time) but convergence can no longer be guaranteed in this case. A good compromise is a partially synchronous scheme: update only conditionally independent pixels at the same time. These pixels form a so-called *coding set* [13] (see Figure 3.5 for an example). The number of coding sets depends on the order of the MRF used as an image model. These techniques are mostly used in parallel implementation but one can also use them on a sequential machine.

A more interesting improvement is to use a non-uniform generation mechanism. For example, at high temperatures, it would be helpful to bias the generation of configurations to favor *large* transitions [83]. Large transitions often help the system to reach equilibrium faster because a single transition of length l can be implemented at a less computational cost than l transitions of length 1.

For *rejection-less methods* [83], the acceptance matrix is incorporated in the generation matrix and all transitions are accepted once they are generated. Considering the Metropolis criterion, the probability of generating ω from η is given by

$$G_{\omega,\eta} = \frac{\min(1, -(U(\eta) - U(\omega)))}{\sum_{\xi} \min(1, -(U(\xi) - U(\omega)))}, \tag{3.13}$$

where ξ differs only in one element from ω. One can prove that the algorithm also converges towards a global optimum. The problem is that after *each* transition, we have to compute *all* $G_{\omega,\eta}$'s.

In [27], the updating order depends on a *local stability measure*. At each step, only the least stable site is changed. The stability of a site is measured by the energy-loss or energy-gain which would be caused by changing the current state of the site. The larger the negative value of this measure, the more confidence we have to change the state of the site. Thus, the sites with strong evidence in favor of a label are visited early. This method is called *Highest Confidence First* (HCF).

3.2.1.3 More on Acceptance Matrixes — The Gibbs Sampler

A more elaborated acceptance matrix has been proposed by D. Geman and S. Geman in [46]. Coupled with an inhomogeneous annealing schedule, it became the most popular SA algorithm known as *Gibbs Sampler*:

Algorithm 3.2.2 (Gibbs Sampler)

① *Set $k = 0$, assign an arbitrary initial configuration ω and let $T = T_0$ be a sufficiently high initial temperature.*

② *For each configuration which differs at most in one element from the current configuration ω (they are denoted by $\mathcal{N}_\omega$), compute the energy $U(\eta)$ ($\eta \in \mathcal{N}_\omega$).*

③ **(Gibbs Sampler)** *From the configurations in $\mathcal{N}_\omega$, a sample is drawn such that η is accepted with probability*

$$\frac{\exp(-U(\eta))}{\sum_{\zeta \in \mathcal{N}_\omega} \exp(-U(\zeta))} \tag{3.14}$$

as the new configuration.

④ *Decrease the temperature: $T = T_{k+1}$ and goto Step ② with $k = k + 1$ until the system is* frozen.

In the case of a two state system such as the Ising model, the Gibbs Sampler is equivalent to the Metropolis algorithm. Notice that the generation matrix is simply $G_{\omega,\eta} = 1$ for all ω and $\eta \in \mathcal{N}_\omega$, 0 otherwise.

3.3 Convergence Study

The SA algorithm obtains a global minimum if, after K transitions (K sufficiently large), the following holds:

$$P(X(K) \in \Omega_{opt}) = 1 \tag{3.15}$$

where Ω_{opt} is the set of globally minimal configurations. In the following sections, we give the conditions on the matrices $A(T)$ and $G(T)$ described by Aarts and van Laarhoven in [83] to ensure the convergence:

$$\lim_{T \searrow 0} (\lim_{K \to \infty} P(X(K) \in \Omega_{opt})) = 1 \tag{3.16}$$

3.3.1 Homogeneous Annealing

In this case, configurations are generated at a fixed temperature until equilibrium is reached. What does it mean from the point of view of the generated sequence of states? They constitute an homogeneous Markov chain with transition matrix given by Equation (3.9). After the equilibrium is reached, the temperature is lowered and configurations are generated at this temperature etc... Thus, the algorithm is described by a sequence of homogeneous Markov chains. Each chain is generated at a fixed value of T and T is decreased in between subsequent chains.

First, conditions are given on the existence of the stationary distribution of the chain, then further conditions are imposed to assure the convergence of the stationary distribution.

Theorem 3.3.1 (Feller) *The stationary distribution q of a finite homogeneous Markov chain exists if the Markov chain is* irreducible *and* aperiodic. *Furthermore, the vector q is uniquely determined by the following equations:*

$$\forall\omega : q_\omega > 0, \sum_\omega q_\omega = 1, \tag{3.17}$$

$$\forall\omega : q_\omega = \sum_\eta q_\eta P_{\eta,\omega}. \tag{3.18}$$

Since $\forall\omega,\eta,T > 0 : A_{\omega,\eta}(T) > 0$ (see Equation (3.11)), and $\forall\omega,\eta(\omega \neq \eta), T > 0 : G_{\omega,\eta}(T) > 0$ ($G(T)$ is uniform), irreducibility is satisfied. To establish aperiodicity, the following theorem is used:

Theorem 3.3.2 *An irreducible Markov chain is aperiodic if the following condition is satisfied:*

$$\forall T > 0 : \exists\omega_T \in \Omega : P_{\omega_T,\omega_T}(T) > 0. \tag{3.19}$$

Thus, for aperiodicity it is sufficient to assume that

$$\forall T > 0 : \exists\omega_T, \eta_T \in \Omega : A_{\omega_T,\eta_T} < 1 \tag{3.20}$$

which is always satisfied by setting, for all $T > 0$, $\omega_T \in \Omega_{opt}, \eta_T \notin \Omega_{opt}$.

Now, further conditions are imposed to ensure the convergence of the stationary distribution (for more details, see [83]):

1. $$\forall\omega,\eta \in \Omega : G_{\eta,\omega} = G_{\omega,\eta} \tag{3.21}$$
2. $$\forall\omega,\eta,\kappa \in \Omega : U(\omega) \leq U(\eta) \leq U(\kappa) \Rightarrow A_{\omega,\kappa}(T) = A_{\omega,\eta}(T)A_{\eta,\kappa}(T) \tag{3.22}$$
3. $$\forall\omega,\eta \in \Omega : U(\omega) \geq U(\eta) \Rightarrow A_{\omega,\eta}(T) = 1 \tag{3.23}$$
4. $$\forall\omega,\eta \in \Omega, T > 0 : U(\omega) < U(\eta) \Rightarrow A_{\omega,\eta}(T) < 1 \tag{3.24}$$
5. $$\forall\omega,\eta \in \Omega : U(\omega) < U(\eta) \Rightarrow \lim_{T\searrow 0} A_{\omega,\eta}(T) = 0 \tag{3.25}$$

It is clear, that the condition 1. is satisfied in our case, since G(T) is uniform, its diagonal is 0 and the other elements are equals. The conditions 2. − 5. are implied by the definition of $A(T)$ in Equation (3.11). These conditions are sufficient but not necessary to ensure the convergence. In this case, the stationary distribution is given by:

$$q_\omega(T) = \frac{\exp(-(U(\omega) - U_{opt})/T)}{\sum_{\eta\in\Omega}\exp(-(U(\omega) - U_{opt})/T)} \tag{3.26}$$

3.3.2 Inhomogeneous Annealing

In this case, configurations are generated such that after each transition, T is lowered. The obtained sequence of configurations constitutes an inhomogeneous Markov chain whose transition matrix $P(k-1,k)$ is defined by:

$$P_{\omega,\eta}(k-1,k) = \begin{cases} G_{\omega,\eta}(T_k)A_{\omega,\eta}(T_k) & \forall\eta\neq\omega \\ 1 - \sum_{\zeta\neq\omega} G_{\omega,\zeta}(T_k)A_{\omega,\zeta}(T_k) & \eta=\omega \end{cases} \tag{3.27}$$

Since T is changed in between subsequent transitions, the conditions of the convergence not only relate to the matrices $G(T_k)$ and $A(T_k)$ but also impose restrictions on the way of changing the control parameter T. The following assumptions are made:

$$1. \quad \lim_{k\to\infty} T_k = 0 \tag{3.28}$$

$$2. \quad T_k \geq T_{k+1},\ k = 0,1\ldots \tag{3.29}$$

Theorem 3.3.3 (Seneta) *An inhomogeneous Markov chain is* weakly ergodic *iff there is a strictly increasing sequence of positive numbers k_l such that*

$$\sum_{l=1}^{\infty}(1 - \tau_1(P(k_l,k_{l+1}))) = \infty; \tag{3.30}$$

where $\tau_1(P)$, the coefficient of ergodicity of P, is defined as

$$\tau_1(P) = 1 - \min_{\omega,\eta}\sum_\zeta \min(P_{\omega,\zeta},P_{\eta,\zeta}). \tag{3.31}$$

Theorem 3.3.4 (Isaacson and Madsen) *An inhomogeneous Markov chain is strongly ergodic if it is weakly ergodic and if for all k there exists a vector $\pi(k)$ such that $\pi(k)$ is an eigenvector with eigenvalue 1 of $P(k-1,k)$, $\sum_\omega \mid \pi_\omega(k) \mid = 1$ and*

$$\sum_{k=1}^{\infty} \sum_{\omega \in \Omega} \mid \pi_\omega(k) - \pi_\omega(k+1) \mid < \infty. \tag{3.32}$$

Moreover, if $\pi = \lim_{k\to\infty} P_{\omega,\eta}(m,k)$, then π is the vector in Definition 1.7.11 satisfying Equation (1.102).

Under the assumptions on $A(T)$ and $G(T)$ described in the previous section , there exists an eigenvector $q(T_k)$ of $P(k-1,k)$ for each $k \geq 0$, namely the stationary distribution of the homogeneous Markov chain. Using Theorem 3.3.4 with $\pi(k) = q(T_k)$, strong ergodicity can be proved by showing that the following conditions are satisfied:

1. The Markov chain is weakly ergodic
2. The $q(T_k)$ satisfy Equation (3.32).

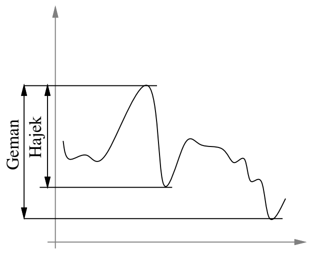

Figure 3.6: *Geman's and Hajek's condition.*

D. Geman and S. Geman were the first researchers to obtain such conditions. The validity of Equation (3.32) is shown in [46] and they proved that

$$T_k \geq \frac{\Gamma}{\ln(k)} \tag{3.33}$$

with

$$\Gamma > \max_{\omega\in\Omega} U(\omega) - \min_{\omega\in\Omega} U(\omega) \tag{3.34}$$

must hold to obtain weak ergodicity.

This condition has been refined by B. Hajek in [58, 59] to derive a necessary and sufficient condition. In Figure 3.6, we compare the meaning of Geman's and Hajek's conditions.

3.3.2.1 Hajek's Necessary and Sufficient Condition

Following [59], we first define the *reachability* of a configuration:

Definition 3.3.1 *A configuration ω is reachable at height H from a configuration η if there is a sequence of configurations $\eta = \zeta_0, \zeta_1, \ldots, \zeta_p = \omega$ such that*

$$\forall k : 0 \leq k < p : \quad G(\zeta_k, \zeta_{k+1}) > 0 \tag{3.35}$$

and

$$\forall k : 0 \leq k \leq p : \quad U(\zeta_k) \leq H \tag{3.36}$$

A configuration ω is said to be a *local minimum* if there is no η with $U(\eta) < U(\omega)$ which is reachable from ω at height $U(\omega)$.

Definition 3.3.2 (Cup) *A cup is a set Υ of configurations such that, for some number H, the following is true:*

$$\forall \omega \in \Upsilon : \quad \Upsilon = \{\eta \in \Omega : \eta \text{ is reachable from } \omega \text{ at height } H\} \tag{3.37}$$

Furthermore, let

$$\Upsilon^{sup} = \max_{\omega \in \Upsilon} U(\omega) \tag{3.38}$$

$$\Upsilon^{inf} = \min_{\omega \in \Upsilon} U(\omega) \tag{3.39}$$

The depth of a cup is defined as $d(\Upsilon) = \Upsilon^{sup} - \Upsilon^{inf}$.

The depth of a local minimum ω is the smallest number $d(\omega)$ such that there is a configuration η with $U(\eta) < U(\omega)$ reachable from ω at height $U(\omega) + d(\omega)$. For a global minimum, the depth is set to ∞. Considering the Markov chain X generated by the SA, we have the following constraints to assure the convergence [59]:

Theorem 3.3.5 (Hajek) *Assume that X is irreducible and for any ω and η, ω is reachable at height H from η if and only if η is reachable from ω at height H (weak reversibility). Assume furthermore that Equation (3.28) and Equation (3.29) hold. Then*

$$\lim_{k \to \infty} P(X_k \in \Omega_{opt}) = 1 \tag{3.40}$$

if and only if

$$\sum_{k=1}^{\infty} \exp\left(-\frac{d^*}{T_k}\right) = \infty \tag{3.41}$$

where d^* is the maximum of depths of all configurations which are local but not global minima.

While Geman's condition says that Γ in Equation (3.33) must be larger than the difference between the maximum and minimum energy, Hajek's condition claims that it is sufficient that Γ is larger than the largest depth of any cup.

3.4 Multi-Temperature Annealing

The main purpose and study of this section is a new Multi-Temperature Annealing (MTA) schedule [73, 76, 117]. In this case, the configurations are generated at different temperatures at different sites. The temperature is then lowered after each transition according to the MTA schedule (see Theorem 3.4.1). More generally, we have the following problem:

Let $\mathcal{S} = \{s_1, s_2, \ldots, s_N\}$ be a set of sites, $\mathcal{V}$ some neighborhood system with cliques $\mathcal{C}$ and X a MRF over these sites with energy function U. We define an annealing scheme where the temperature T depends on the iteration k and on the cliques C. Let $\oslash$ denotes the following operation:

$$U(\omega) \oslash T(k,C) = \sum_{C\in\mathcal{C}} \frac{V_C(\omega)}{T(k,C)} \tag{3.42}$$

$$P(X=\omega) = \pi_{T(k,C)}(\omega) = \frac{\exp(-U(\omega)\oslash T(k,C))}{Z} \tag{3.43}$$

Let us suppose that the sites are visited for updating in the order $\{n_1, n_2, \ldots\} \subset \mathcal{S}$. The resulting stochastic process is denoted by $\{X(k), k = 0, 1, 2, \ldots\}$, where $X(0)$ is the initial configuration. $X(k)$ is an inhomogeneous Markov chain with transition matrix:

$$P_{\omega,\eta}(k-1,k) = \begin{cases} G_{\omega,\eta}(T(k,C))A_{\omega,\eta}(T(k,C)) & \forall\eta\neq\omega \\ 1-\sum_{\zeta\neq\omega} G_{\omega,\zeta}(T(k,C))A_{\omega,\zeta}(T(k,C)) & \eta=\omega \end{cases} \tag{3.44}$$

Considering the Gibbs sampler, the generation matrix $G_{\omega,\eta}(T(k,C))$ and acceptation matrix $A_{\omega,\eta}(T(k,C))$ are given by:

$$G_{\omega,\eta}(T(k,C)) = G_{\omega,\eta}(k) = \begin{cases} 1, & \text{if } \eta = \omega|_{\omega_{n_k}=\lambda} \text{ for some} \lambda\in\Lambda \\ 0, & \text{otherwise} \end{cases} \tag{3.45}$$

$$A_{\omega,\eta}(T(k,C)) = \pi_{T(k,C)}(X_{n_k} = \omega_{n_k} \mid X_s = \omega_s, s\neq n_k) \tag{3.46}$$

Notice that the acceptance is governed by the *local* characteristics. $\pi_{T(k,C)}(X_{n_k} = \omega_{n_k} \mid X_s = \omega_s, s\neq n_k)$ has a slightly different meaning than $\pi_{T(k,C)}(\omega)$ in Equation (3.43):

$$\pi_{T(k,C)}(X_s = \omega_s \mid X_r = \omega_r, s\neq r) = \frac{1}{Z_s}\exp\left(-\frac{\sum_{C\in\mathcal{C}:s\in C} V_C(\omega)}{T(k,C)}\right) \tag{3.47}$$

$$\text{with } Z_s = \sum_{\lambda\in\Lambda}\exp\left(-\frac{\sum_{C\in\mathcal{C}:s\in C} V_C(\omega|_{\omega_s=\lambda})}{T(k,C)}\right) \tag{3.48}$$

The transition matrix at time k is then of the following form:

$$P_{\omega,\eta}(k) = \begin{cases} \pi_{T(k,C)}(X_{n_k} = \eta_{n_k} \mid X_s = \eta_s, s \neq n_k), & \text{if } \eta = \omega|_{\omega_{n_k}=\lambda} \text{ for some } \lambda \in \Lambda \\ 0, & \text{otherwise} \end{cases} \tag{3.49}$$

Let Ω_{opt} be the set of globally optimal configurations:

$$\Omega_{opt} = \{\omega \in \Omega : U(\omega) = \min_{\eta \in \Omega} U(\eta)\} \tag{3.50}$$

Let π_0 be the uniform distribution on Ω_{opt}, and define:

$$\begin{aligned} U^{sup} &= \max_{\omega \in \Omega} U(\omega), && (3.51) \\ U^{inf} &= \min_{\omega \in \Omega} U(\omega), && (3.52) \\ \text{and } \Delta &= U^{sup} - U^{inf}. && (3.53) \end{aligned}$$

Let us examine the decomposition of $U(\omega) \oslash T(k, C)$ defined in Equation (3.42). Let $\omega' \in \Omega_{opt}$ be a *globally* optimal configuration ($U(\omega') = U^{inf}$). Furthermore, let $\omega \in \Omega \setminus \Omega_{opt}$ be any other non-optimal configuration. Obviously, $U(\omega) - U(\omega') > 0$. In the case of a classical annealing, dividing by a constant temperature does not change this relation ($\forall k$: $(U(\omega) - U(\omega'))/T_k$ is still positive). But it is not necessarily true that $(U(\omega) - U(\omega')) \oslash T(k, C)$ is also positive! Because choosing sufficiently small temperatures for the cliques where ω'_C is *locally* not optimal (i.e. strengthening the cliques where $V_C(\omega) - V_C(\omega') < 0$) and choosing sufficiently high temperatures for the cliques where ω'_C is *locally* optimal (i.e. weakening the cliques where $V_C(\omega) - V_C(\omega') \geq 0$), we obtain $(U(\omega) - U(\omega')) \oslash T(k, C) < 0$, meaning that ω' is no longer *globally* optimal with respect to $U \oslash T(k, C)$.

Thus, we have to impose further conditions on the temperature to assure the convergence towards the *original* optimum of U. First, let us examine the decomposition over the cliques of $U(\omega) - U(\eta)$ for arbitrary ω and η, $\omega \neq \eta$:

$$U(\omega) - U(\eta) = \sum_{C \in \mathcal{C}} (V_C(\omega) - V_C(\eta)). \tag{3.54}$$

Indeed, there may be negative and positive members in the decomposition. According to this fact, we have the following subsums:

$$\begin{aligned} &\sum_{C \in \mathcal{C}} (V_C(\omega) - V_C(\eta)) \\ = &\underbrace{\sum_{C \in \mathcal{C}:(V_C(\omega)-V_C(\eta))<0} (V_C(\omega) - V_C(\eta))}_{\Sigma^-(\omega,\eta)} + \underbrace{\sum_{C \in \mathcal{C}:(V_C(\omega)-V_C(\eta))\geq 0} (V_C(\omega) - V_C(\eta))}_{\Sigma^+(\omega,\eta)}. \end{aligned} \tag{3.55}$$

Now, let us examine Δ defined in Equation (3.53). If we want to decompose Δ as defined above, we have to choose some configuration ω' with a maximum energy (i.e. $U(\omega') = U^{sup}$) and another configuration ω'' with a minimum energy (i.e. $U(\omega'') = U^{inf}$). Obviously, there may be more than one decomposition depending on the number of globally optimal configurations ($| \Omega_{opt} |$) and the number of configurations with maximal global energy ($| \Omega_{sup} |$). Thus, the decomposition of Δ for a given (ω', ω'') is of the following form:

$$\Delta = \Sigma^{-}(\omega', \omega'') + \Sigma^{+}(\omega', \omega'') \tag{3.56}$$

Furthermore, let us define Σ^{+}_{Δ} as:

$$\Sigma^{+}_{\Delta} = \min_{\substack{\omega' \in \Omega_{sup} \\ \omega'' \in \Omega_{opt}}} \Sigma^{+}(\omega', \omega''). \tag{3.57}$$

Obviously $\Delta \leq \Sigma^{+}_{\Delta}$. The following theorem gives an annealing schedule, basically the same as in [46]. *However, the temperature here is a function of k and $C \in \mathcal{C}$.*

Theorem 3.4.1 (Multi-Temperature Annealing) *Assume that there exists an integer $\kappa \geq N$ such that for every $k = 0, 1, 2, \ldots$, $\mathcal{S} \subseteq \{n_{k+1}, n_{k+2}, \ldots, n_{k+\kappa}\}$. For all $C \in \mathcal{C}$, let $T(k, C)$ be any decreasing sequence of temperatures in k for which*

(i) $\lim_{k\to\infty} T(k, C) = 0$.
Let us denote respectively by T_k^{inf} and T_k^{sup} the maximum and minimum of the temperature function at k ($\forall C \in \mathcal{C}$: $T_k^{inf} \leq T(k, C) \leq T_k^{sup}$).

(ii) *For all $k \geq k_0$, for some integer $k_0 \geq 2$: $T_k^{inf} \geq N\Sigma^{+}_{\Delta} / \ln(k)$.*

(iii) *If $\Sigma^{-}(\omega, \omega') \neq 0$ for some $\omega \in \Omega \setminus \Omega_{opt}$, $\omega' \in \Omega_{opt}$ then a further condition must be imposed:*
For all k: $\frac{T_k^{sup} - T_k^{inf}}{T_k^{inf}} \leq R$ with

$$R = \min_{\substack{\omega \in \Omega \setminus \Omega_{opt} \\ \omega' \in \Omega_{opt} \\ \Sigma^{-}(\omega, \omega') \neq 0}} \frac{U(\omega) - U^{inf}}{| \Sigma^{-}(\omega, \omega') |}. \tag{3.58}$$

Then for any starting configuration $\eta \in \Omega$ and for every $\omega \in \Omega$:

$$\lim_{k\to\infty} P(X(k) = \omega \mid X(0) = \eta) = \pi_0(\omega). \tag{3.59}$$

The proof of this theorem appears in Appendix 3.A.

Remarks:

1. In practice, we cannot determine R and Σ^+_Δ, as we cannot compute Δ either.
2. Considering Σ^+_Δ in condition 3.4.1/ii, we have the same problem as in the case of a classical annealing. The only difference is that in a classical annealing, we have Δ instead of Σ^+_Δ. Consequently, the same solutions may be used: an exponential schedule with a sufficiently high initial temperature.
3. The factor R is more interesting. We propose herein two possibilities which can be used for practical implementations of the method: Either we choose a sufficiently small interval $[T_0^{inf}, T_0^{sup}]$ and suppose that it satisfies the condition 3.4.1/iii (we have used this technique in the simulations), or we use a more strict but easily verifiable condition [117] instead of condition 3.4.1/iii, namely:
$$\lim_{k\to\infty} \frac{T_k^{sup} - T_k^{inf}}{T_k^{inf}} = 0. \tag{3.60}$$
4. What happens if $\Sigma^-(\omega, \omega')$ is zero for all ω and ω' in condition 3.4.1/iii and thus R is not defined? This is the best case because it means that all *globally* optimal configurations are also *locally* optimal. That is we have no restriction on the interval $[T_k^{inf}, T_k^{sup}]$, thus any *local* temperature schedule satisfying conditions 3.4.1/i–3.4.1/ii is good.

To illustrate a MTA schedule, we have done a computer simulation with the following example:

Example 3.4.1 Consider the cost function:
$$U(X) = \cos(X_1) + (\arctan(X_1) - \sin(X_2)) - \arctan(X_2)$$
where $X = (X_1, X_2)$ is a MRF with cliques $\{X_1\}$, $\{X_2\}$ and $\{X_1, X_2\}$. The state space is simply $\{0, 1, \ldots, 9\}$. Since the energy function is simple, we can compute it for each possible configuration (cf. Table 3.1). We obtain $R = 0.145387$, $\Delta = 4.542564$ and $\Sigma^+_\Delta = 4.542564$. Now, let us define a temperature schedule according to Theorem 3.4.1:
$$\begin{aligned} T(k, \{X_1\}) &= \frac{1 \cdot 5 \cdot N}{\ln(k)}, \\ T(k, \{X_1, X_2\}) &= \frac{1.1 \cdot 5 \cdot N}{\ln(k)}, \\ T(k, \{X_2\}) &= \frac{1.14 \cdot 5 \cdot N}{\ln(k)}, \end{aligned}$$

yielding the following energy function:

$$U_{MTA}(X,k) = \left(\cos(X_1) + \frac{\arctan(X_1) - \sin(X_2)}{1.1} + \frac{\arctan(X_2)}{1.14}\right)\frac{\ln(k)}{5 \cdot N},$$

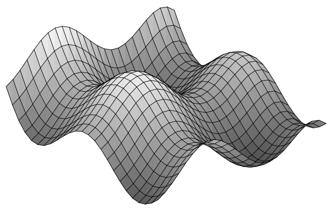

Figure 3.7: *Landscape of the energy function with continuous state space.*

which is nothing else but a conventional annealing with the modified energy function

$$U'(X) = \cos(X_1) + \frac{\arctan(X_1) - \sin(X_2)}{1.1} + \frac{\arctan(X_2)}{1.14}.$$

The values of U' have also been computed as shown in Table 3.2. Clearly, we obtain the same minimum as for the original function U, that is $X = (3, 8)$. Notice, however, that the obtained minimum value is not the same! Clearly, the MTA schedule modifies the original energy function, but the modified function has the same minima. This is what we have proved in Theorem 3.4.1.

$X_1 \backslash X_2$	0	1	2	3	4	5	6	7	8	9
0	1.00	-0.63	-1.02	-0.39	0.43	0.59	-0.13	-1.09	-1.44	-0.87
1	1.33	-0.30	-0.69	-0.06	0.76	0.91	0.20	-0.76	-1.11	-0.55
2	0.69	-0.94	-1.33	-0.70	0.12	0.28	-0.44	-1.39	-1.74	-1.18
3	0.26	-1.37	-1.76	-1.13	-0.31	-0.16	-0.87	-1.83	**-2.18**	-1.61
4	0.67	-0.95	-1.34	-0.72	0.10	0.26	-0.45	-1.41	-1.76	-1.20
5	1.66	0.03	-0.36	0.27	1.09	1.24	0.53	-0.43	-0.78	-0.22
6	2.37	0.74	0.35	0.98	1.80	1.95	1.24	0.28	-0.07	0.49
7	2.18	0.56	0.17	0.79	1.61	1.77	1.06	0.10	-0.25	0.31
8	1.30	-0.33	-0.72	-0.09	0.73	0.89	0.17	-0.78	-1.13	-0.57
9	0.55	-1.08	-1.47	-0.84	-0.02	0.13	-0.58	-1.54	-1.89	-1.32

Table 3.1: *Original energy function U.*

$X_1 \backslash X_2$	0	1	2	3	4	5	6	7	8	9
0	1.00	-0.45	-0.80	-0.22	0.53	0.67	0.02	-0.85	-1.17	-0.66
1	1.25	-0.20	-0.54	0.03	0.78	0.92	0.28	-0.60	-0.91	-0.40
2	0.59	-0.86	-1.21	-0.63	0.12	0.26	-0.39	-1.26	-1.58	-1.07
3	0.15	-1.31	-1.65	-1.08	-0.33	-0.19	-0.83	-1.71	**-2.02**	-1.51
4	0.55	-0.90	-1.25	-0.67	0.08	0.22	-0.43	-1.30	-1.62	-1.10
5	1.53	0.08	-0.27	0.31	1.06	1.20	0.55	-0.32	-0.64	-0.12
6	2.24	0.78	0.44	1.01	1.76	1.91	1.26	0.39	0.07	0.58
7	2.05	0.60	0.26	0.83	1.58	1.72	1.07	0.20	-0.12	0.40
8	1.17	-0.28	-0.63	-0.05	0.69	0.84	0.19	-0.68	-1.00	-0.49
9	0.42	-1.04	-1.38	-0.81	-0.06	0.08	-0.56	-1.43	-1.75	-1.24

Table 3.2: *Modified energy function U'.*

3.4.1 Application to Hierarchical Markov Models

Hierarchical models usually require much more communication per pixel than monogrid ones. This is why classical annealing schemes are too slow even on a parallel machine to minimize the energy associated with such a model. However, taking benefit of the pyramidal structure of the model, we can define a MTA scheme, which consists of associating higher temperatures to higher levels, in order to be less sensitive to local minima at coarser grids (see Figure 3.8). For the cliques siting between two levels, we use either the temperature of the lower level or the higher level (but once chosen, we always keep the same level throughout the algorithm). In Section 3.7, we will compare the MTA schedule with a classical inhomogeneous scheme. Tests have shown that the MTA algorithm converges much faster than the classical SA.

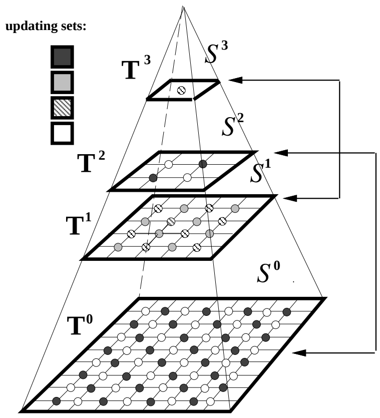

Figure 3.8: *Relaxation scheme on the pyramid.*

3.5 Deterministic Relaxation

SA algorithms reach a global minimum but they require a large amount of computation. On the other hand, a global optimum is obtained only theoretically. In practice, we always implement an *approximation* of the SA and the convergence towards the global optimum is no longer guaranteed.

To speed up the convergence, many authors propose *deterministic* algorithms [13, 15, 79, 78, 80]. While the essence of every *stochastic* relaxation is that transitions with energy *increase* are permitted, *deterministic* relaxation allows only transitions with energy *decrease*.

Let us begin the discussion with the most popular deterministic algorithm: the *Iterated Conditional Modes* (ICM) [13].

3.5.1 Iterated Conditional Modes (ICM)

If we have a reasonably good initial configuration then an extremely rapid convergence can be obtained by the ICM method proposed by Besag in [13]. The quality of the final result strongly depends on the initialization since ICM realizes only a descent in the nearest energy-valley. Of course, the obtained minimum is only *local* but convergence towards this minimum is obtained usually in a few iterations (less then 10 in our experiments).

Algorithm 3.5.1 (ICM)

① *Start at a "good" initial configuration* ω^0 *and set* $k = 0$.

② *For each configuration which differs at most in one element from the current configuration* ω^k *(they are denoted by* $\mathcal{N}_{\omega^k}$*), compute the energy* $U(\eta)$ $(\eta \in \mathcal{N}_{\omega^k})$.

③ *From the configurations in* $\mathcal{N}_{\omega^k}$*, select the one which has a minimal energy:*

$$\omega^{k+1} = \arg \min_{\eta \in \mathcal{N}_{\omega^k}} U(\eta). \tag{3.61}$$

④ *Goto Step ② with* $k = k + 1$ *until convergence is obtained (for example, the energy change is less than a certain threshold).*

Notice that in the ICM algorithm there is no temperature parameter and thus there is no annealing. On the other hand, Step ③ is nothing else but the acceptance rule of the Gibbs Sampler (see Algorithm 3.2.2) at $T = 0$. Thus, the ICM algorithm corresponds to a purely deterministic "frozen" Gibbs Sampler.

The initialization method depends on the problem which we are trying to solve. For image labeling, one normally adopts the conventional *maximum likelihood* estimator which ignores spatial dependence of one pixel on the others. Other examples can be found in [13].

3.5.2 Graduated Non-Convexity (GNC)

The idea behind GNC [15] is to approximate the *non-convex* energy function U by a new function $U^\star$ which is *convex* and hence can only have one minimum. In the simplest case, this minimum may also be the *global minimum* of the original function U. Generally, we need a *sequence* of functions U^p $(0 \leq p \leq 1)$ such that $U^0 = U$ and $U^1 = U^\star$. In between, U^p changes in a continuous way, between U and $U^\star$. The algorithm itself consists of minimize the sequence U^p $(p = 1 \rightarrow p = 0)$ using the result of one optimization as the starting point for the next. The main inconvenient of the method is that there is no exact formula how to choose a convex approximation. It depends on the original function U. Some examples can be found in [15].

Algorithm 3.5.2 (GNC)

① *Define a convex approximation $U^\star$ of U. Set up a sequence of approximations U^{p_i}, $\forall i = 1 \dots P: \ 0 \leq p_i \leq 1$ such that $U^0 = U$ and $U^1 = U^\star$. Initialize $i = 1$.*

② *Find the minimum $\hat{\omega}^i$ of U^{p_i} (by a direct descent or gradient descent method, for example).*

③ *Goto Step ② with $\hat{\omega}^i$ as the initial configuration and $i = i + 1$ until $i < P$.*

3.5.3 Deterministic Pseudo Annealing (DPA)

DPA is a GNC-like algorithm proposed by M. Berthod *et al.* in [10]. It is also related to relaxation labeling algorithms [67, 38]. The basic idea is to extend the probability of a discrete labeling of pixels in an image to a merit function defined on continuous labelings which is a polynomial with non-negative coefficients. Under certain constraints, the only extrema of this function is a discrete labeling. DPA consists of changing these constraints to convexify the merit function, find its global maximum, and then track down the solution until the original constraints are restored yielding an optimal discrete labeling of the original problem.

Let us consider the energy function of a discrete labeling ω:

$$U(\omega) = \sum_{C \in \mathcal{C}} V_C(\omega). \tag{3.62}$$

To obtain optimal labeling, we have to minimize this function function, or equivalently, maximize the negative energy:

$$-U(\omega) = \sum_{C \in \mathcal{C}} -V_C(\omega) = \sum_{C \in \mathcal{C}} W_C(\omega). \tag{3.63}$$

It is possible to shift all W_C's so that they all become positive, without changing the solution. Now, this combinatorial optimization problem is transformed into a maximization problem in a compact subset of $\Re^{NL}$ (N is the number of sites and L is the number of elements in the common state space). Following [10], we define the following real function:

$$f(X) = \sum_{C \in \mathcal{C}} \sum_{\omega \in \Omega_C} W_C(\omega) \prod_{i=1}^{\deg(C)} x_{C_i, \omega_{C_i}}, \tag{3.64}$$

where C_i denotes the i^{th} site of clique C and Ω_C denotes the set of all possible labelings of the sites of clique C. Indeed, f is a polynomial in $x_{i,j}$'s, its degree is the maximum degree of the cliques ($\deg(\mathcal{C})$). If f is restricted to a compact subset $\mathcal{P}^{NL}$ of $\Re^{NL}$:

$$\forall i, j: \quad x_{i,j} \geq 0 \text{ and } \forall i: \sum_{j=1}^{L} x_{i,j} = 1. \tag{3.65}$$

The maximum of f on $\mathcal{P}^{NL}$ is on the border:

$$\forall i \exists j: \quad x_{i,j} = 1 \text{ and } \forall k \neq j: \quad x_{i,k} = 0. \tag{3.66}$$

Thus any maxima on $\mathcal{P}^{NL}$ directly yields a discrete labeling. To find the global maximum on $\mathcal{P}^{NL}$, DPA proceeds in the following way: Maximize f on a subset $\mathcal{Q}^{NL,d}$:

$$\forall i \exists j: \quad x_{i,j} = 1 \text{ and } \forall k \neq j: \quad x_{i,k} = 0 \tag{3.67}$$

on which it is concave and track down the maximum by slowly restoring the original subset $\mathcal{P}^{NL}$. As claimed in [10], one can prove that f has a unique maximum on $\mathcal{Q}^{NL,d}$. Maximization is performed by the *iterative power method.*

Algorithm 3.5.3 (DPA)

① *Set $d = 2$ and initialize X by some X_0.*

② *Find $\hat{X}$ which maximizes f on $\mathcal{Q}^{NL,d}$ using the iterative power method:*

$$X_{n+1} = (\nabla f(X_n))^{\frac{1}{d-1}} \qquad n = 0, 1, 2, \ldots \tag{3.68}$$

③ *Decrease d by some quantity and project $\hat{X}$ on the new $\mathcal{Q}^{NL,d}$. Goto Step ② with X_0 equals to the projection of $\hat{X}$ until $d > 1$.*

④ *For each site i, select the label j for which $x_{i,j} = 1$.*

This iterative decrease of d can be compared, up to a point to a cooling schedule, or better to a Graduated Non-Convexity strategy [15].

Geometrically, Step ② simply means that at each iteration, we select on the pseudo-sphere of degree d the point where the normal is parallel to the gradient of f. This cannot be applied when $d = 1$, as claimed in [10], the procedure must stop for some d slightly larger than 1. The convergence of the algorithm has not been proved but experiments show [80, 75, 11] that on real problems a very good solution is reached.

3.5.4 Game Strategy Annealing (GSA)

Based on the game theory, Liu-Yu proposes a relaxation scheme called *Game Strategy Approach* [87]. Herein, we give a slightly modified version of the original algorithm [113]. The algorithm is essentially a Metropolis algorithm with deterministic acceptance rule and modified generation mechanism depending on *local energies*[2].

Algorithm 3.5.4 (GSA)

① *Choose an initial configuration* ω^0 *and set* $k = 0$.

② *For each element* ω_s^k $(s \in \mathcal{S})$ *of* ω^k, *select a state* $\omega'_s \in \Lambda$ *such that*

$$\omega'_s = \arg \min_{\lambda \in \Lambda \setminus \{\omega_s^k\}} U_s(\lambda) \tag{3.69}$$

where $U_s(\lambda)$ *is the local energy in* s *when* s *is in state* λ. *That is, we select at each site the state with minimal energy,* locally.

③ *Accept* ω'_s *at* s *as the new state with probability* α *if* $U_s(\omega'_s) < U_s(\omega_s^k)$. *More precisely:*

$$\omega_s^{k+1} = \begin{cases} \omega'_s & \text{if } U_s(\omega'_s) < U_s(\omega_s^k) \text{ and } \alpha \leq \exp(-U(\omega^k) - U(\omega^k|_{\omega_s^k = \omega'_s})) \\ \omega_s^k & \text{otherwise} \end{cases} \tag{3.70}$$

where α *is constant chosen at the beginning of the algorithm.*

④ *Goto Step* ② *with* $k = k + 1$ *until convergence is obtained.*

Notice that in the algorithm, there is no temperature parameter thus there is no annealing. However, we can use an annealing schedule in Equation (3.70). According to

[2] The local energy can be problem-dependent. For image processing, it may be the sum of potentials over the cliques containing a site s.

our experience, annealing may speed up the convergence of GSA (see Section 3.7).

3.5.5 Modified Metropolis Dynamics (MMD)

Here, we propose a pseudo-stochastic variation of the Metropolis Dynamics [79, 78, 80]. At high temperature, the behavior of our algorithm is similar to the stochastic techniques. However, if the temperature is less than a certain threshold, it becomes deterministic. The "length" of the "pseudo-stochastic" phase is controlled by a constant threshold used in the modified Dynamics. The difference between the Metropolis dynamics and our approach is the choice of ξ in Step ③ of Algorithm 3.2.1. For the original method, ξ is chosen randomly at each iteration, however for our algorithm, ξ is a constant threshold, say α, chosen at the beginning of the algorithm. This simply means that the jump to η is allowed if this does not increase *excessively* the energy. The threshold α controls this increasing of energy.

Algorithm 3.5.5 (MMD)

① *Pick up randomly an initial configuration ω^0, with $k = 0$ and $T = T_0$.*

② *Using a uniform distribution, pick up a global state η which differs only in one element from ω^k.*

③ **(Modified Metropolis Dynamics)** *Compute $\Delta U = U(\eta) - U(\omega)$ and accept η according to the following rule:*

$$\omega^{k+1} = \begin{cases} \eta & \text{if } \Delta U \leq 0, \\ \eta & \text{if } \Delta U > 0 \text{ and } \ln(\alpha) \leq \left(-\frac{\Delta U}{T}\right), \\ \omega^k & \text{otherwise} \end{cases} \tag{3.71}$$

where α is a constant threshold ($\alpha \in (0,1)$), chosen at the beginning of the algorithm.

④ *Decrease the temperature $T = T_{k+1}$ and goto Step ② until convergence is obtained (ΔU less than a certain threshold, for example).*

The MMD algorithm is much faster than the original Metropolis as we will see in Section 3.7. Because for MMD, in Step ②, we have only to compute $\Delta U/T$ and compare it to $\ln(\alpha)$ which is a constant computed at the beginning of the algorithm. However for the original Metropolis dynamics, we have to compute $\exp(-\Delta U/T)$ at each iteration since it is compared to a random value which is not constant. The behavior of our algorithm is as follows. We can separate two phases:

Pseudo-Stochastic Phase: At high temperature, the energy increase is permitted. The behavior of our algorithm is similar to the stochastic techniques.

Deterministic Phase: If the temperature is less than a certain threshold then only the jumps to configurations of lower energy are allowed. The behavior is thus similar to the deterministic algorithms and it converges to a local minimum.

The initialization is not as crucial as for the ICM algorithm because the *pseudo-stochastic phase* results in a good initialization for the *deterministic phase*. There is no explicit formula to get the threshold α. In practice, α is determined by an ad-hoc way depending on the landscape of the energy function. If it is smooth enough, a shorter "stochastic" phase is sufficient thus α is chosen nearly equal to one.

The following theorem provides a more precise characterization of these phases.

Theorem 3.5.1 (MMD) *For any $\alpha \in (0,1)$, there exists a temperature threshold*

$$T_\alpha = -\frac{\Delta U_{min}}{\ln(\alpha)} \tag{3.72}$$

$$\text{where } \Delta U_{min} = \min_{\substack{\omega, \eta \in \Omega \\ U(\omega) \neq U(\eta)}} | U(\omega) - U(\eta) | \tag{3.73}$$

such that if $T_k < T_\alpha$ then only configurations with lower energy will be accepted and thus the algorithm converges towards a local minimum.

If $T_k = \Gamma / \ln(k)$ then $T_k < T_\alpha$ if and only if $k > K_\alpha$ where K_α is a threshold given by:

$$K_\alpha = \exp\left(-\frac{\Gamma \cdot \ln(\alpha)}{\Delta U_{min}}\right). \tag{3.74}$$

In other words, after K_α iterations, the MMD algorithm enters the *deterministic phase* accepting only configurations with an energy decrease. The proof of the above theorem can be found in Appendix 3.B.

3.6 Parallelization Techniques

In the previous section, we have discussed a variety of deterministic algorithms proposed by different authors as an alternative to minimize non-convex functions. In this section, we deal with parallelization techniques adapted for stochastic as well as for deterministic methods.

3.6.1 Data Parallelism

As a natural parallelization method in a lot of image processing problem, we have already mentioned the coding scheme [13] in Section 3.2.1.2. It consists of constructing *coding sets* such that pixels belonging to the same set are conditionally independent, thus they can be updated at the same time (see Figure 3.5 for an example). The main advantage of this technique is that it does not violate the convergence.

Another interesting method has been proposed by Azencott in [4]: At each iteration k, each site belongs to the active set A_k with probability τ ($\tau \in (0,1]$ is fixed). The updating is then carried out simultaneously at the active sites in A_k. By convention, $\tau = 0$ denotes the *sequential* algorithm and $\tau = 1$ corresponds to the fully parallel scheme where *all* sites are updated at the same time. For $\tau \in (0,1)$, Trouvé has proved in [110] that using an appropriate cooling schedule (see Section 3.3), the generated Markov chain converges:

$$\lim_{k \to \infty} P(X_k = \omega) = \pi_\tau(\omega). \tag{3.75}$$

The main result of Trouvé is that partially parallelized algorithms are *equivalent* with respect to their limiting distribution:

$$\forall \tau\colon 0 < \tau < 1\colon \quad \pi_\tau \equiv \Pi. \tag{3.76}$$

However, it is *not* shown whether $\Pi \equiv \pi_0$ (π_0 denotes the stationary distribution of the sequential annealing which is known to coincide with the uniform distribution over the optimal configurations) but the experimental study in [41] seems to confirm this equivalence. On the other hand, many examples can be constructed where π_1 is *not* equivalent to π_0.

The parallelization schemes described here assume that the configuration space can be partitioned (in image processing, it is usually true). On the other hand, the optimization algorithm itself is still sequential since transitions are carried out one after the other which is typically a sequential process. We have only changed the generation mechanism. In the followings, we will study parallel implementations where Markov chains are generated simultaneously.

3.6.2 Parallel Simulated Annealing

Essentially, there are two approaches [83, 5]. The first one, called *systolic algorithm*, aims at generating multiple Markov chains with possible interactions between them. In the other approach, called *clustered algorithm*, all processors are used to generate cooperatively the same Markov chain.

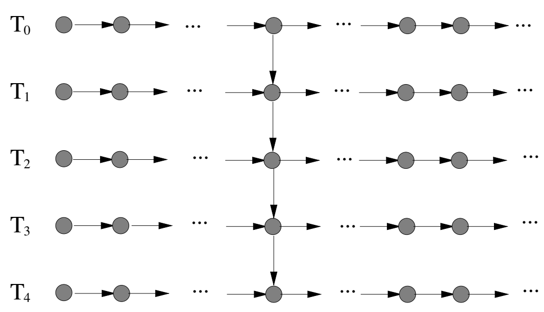

Figure 3.9: *Systolic parallelization scheme.*

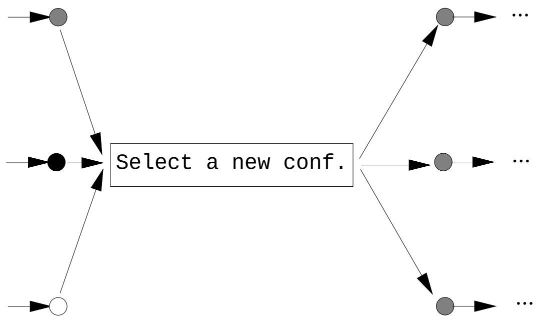

Figure 3.10: *Clustered parallelization scheme.*

3.6.2.1 Systolic Algorithm

The basic idea is to generate n Markov chains simultaneously [83, 5, 53] (cf. Figure 3.9). Obviously, if the chains are independent and the temperature T is fixed (in Figure 3.9, $T_0 = T_1 = \cdots = T_4$), they generate the *same* Markov chain. According to the convergence results presented in Section 3.3, after an infinite number of iterations we obtain n realizations of the same chain. One possibility is to select the optimum among the n realizations. In [54], it has been shown that if the stationary distribution of the sequential homogeneous algorithm is denoted by $q(T)$ then, using n processors, the stationary distribution of the above defined parallel homogeneous algorithm is given by $q(T/n)$. Since the convergence rate of SA increases exponentially when T goes to 0, it is obvious that using n processors may considerably speed up the convergence.

A more general scheme is to allow communications between the processors at regular time intervals. The interactions may consist of selecting the best configuration among the n configurations or one can use a probabilistic rule (the acceptance rule of the Gibbs Sampler in Equation (3.14), for example). Surprisingly, this scheme is *asymptotically less efficient* than the independent scheme if each processor is executing *the same* annealing algorithm. As claimed in [4], performing interactions between n processors executing the same annealing is only a waste of computing time. It might as well be replaced by a single interaction at the end to select the best final configuration. The mathematical study of this method is provided in [53, 6] and experimental results can be found in [53, 54]. We mention here a conjecture from [54]:

Conjecture 3.6.1 (Graffigne) *Consider n processors generating the same homogeneous Markov chain at temperature T interacting after $l, 2l, 3l \ldots$ iterations. The resulting stochastic process is a nonhomogeneous Markov chain $X_k = (X_k^1, X_k^2, \ldots, X_k^n)$. If l is sufficiently large to allow reaching each configuration in maximum l transitions then the asymptotic law of $X^n(T)$ is close to $q(T/n)$. This means that there is no need to add extra interactions before the last one.*

Finally, let us discuss a more interesting scheme. The approach is exactly the same as before but the temperature now varies for each processor. The processors generate different Markov chains at different temperatures. Usually, the first processor is set to a high temperature, the last processor is set to a temperature close to 0 and the other processors are set to intermediate temperatures uniformly distributed between the highest and lowest ones (cf. Figure 3.9 with $T_0 > T_1 > \cdots > T_4$). The behavior of the algorithm is as follows: The first processor, at a high temperature, will randomly explore the energy landscape with large moves. The lower temperatures allows to investigate a selected energy-valley and the last processor, with temperature close to zero, will find the local minima. The convergence of the algorithm has not been proved but the experimental results presented in [54] are very promising.

We have used the idea of performing relaxation at different temperatures in our Multi-Temperature Annealing schedule (see Section 3.4) but, in our case, the convergence has been proved (see Appendix 3.A).

3.6.2.2 Clustered Algorithm

Here, all processors are used to carry out a transition of the same Markov chain (see Figure 3.10). Then the new configuration will be selected among the n configurations according to a deterministic or probabilistic rule as in the previous section. In [22], a detailed mathematical study is provided confirming the convergence of the method.

3.6.3 Parallel Multiscale Algorithms

Herein, we present some parallelization schemes for the multiscale relaxation algorithm described in Algorithm 2.5.1. The most simple method is to use a *data parallel* relaxation algorithm (see Section 3.6.1) at each level. In this case, the convergence of SA towards a global minimum is guaranteed. However, the levels are handled sequentially. Full parallelization is not a trivial task since the algorithm is intrinsically sequential (see Figure 2.8): we need the result of the coarser level to initialize the level below it. Many heuristics have been proposed to introduce additional parallelism in the pyramid. The common problem of these methods is that convergence is no longer guaranteed. Nevertheless, experimental results seem to give a reasonably support of the convergence.

In [91, 62] a parallel inter-level strategy has been proposed, similar to Graffigne's parallelized Markov chain approach [54, 53]. The algorithm consists of running a (possibly data parallel) relaxation algorithm at each level of the pyramid with different initial temperatures (the highest temperature is assigned to the coarsest grid). In regular time intervals, the coarse grids transmit a small block of labels (an *interaction block*) to the level below it. The block is accepted at the finer level if its energy is lower. The energies can be compared directly (after projection of the interaction block into the finer grid) due to the consistent definition of the energy functions at higher levels (they are all related to the energy function of the finest level as explained in Section 2.5).

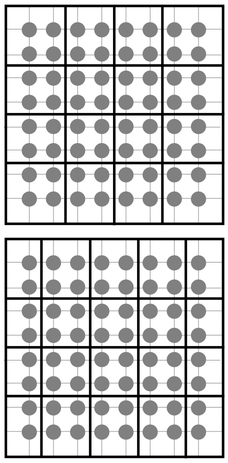

Figure 3.11: *Multiple partitioning in a multiscale model.*

Another scheme has been proposed in [92]. Considering the partition of the original grid $\mathcal{S}$ reported in Algorithm 2.5.1, we may notice that the block partitioning is not unique at a given resolution. As shown in Figure 3.11, partitions can be obtained by considering successive shifts of the initial block partition along the horizontal and vertical directions (we have exactly $(wh)^l$ different partitionings at level l, as pointed out in [92]). Taking benefit of these different partitionings, we can define a parallel scheme: For each partitioning at a given level, we associate a relaxation algorithm. Thus we obtain $(wh)^l$ algorithm running in parallel. At the convergence, the configuration of lowest energy is selected among the $(wh)^l$ results.

For the ICM algorithm [13] (see Algorithm 3.5.1), a *multi-initialization* method has been proposed in [92]. It consists of running multiple ICM algorithm at coarser levels with different initial configurations. The next level is initialized with the configuration of minimal energy.

3.6.4 A Parallel Hierarchical Scheme

From the optimization view-point, there is no difference between a hierarchical and monogrid MRF model: We have a non-convex energy function defined over partitionable configurations. Consequently, the same parallelization techniques may be used. However, there is a special setup for hierarchical models, namely the Multi-Temperature Annealing coupled with a coding parallelization scheme. In Section 3.4.1,

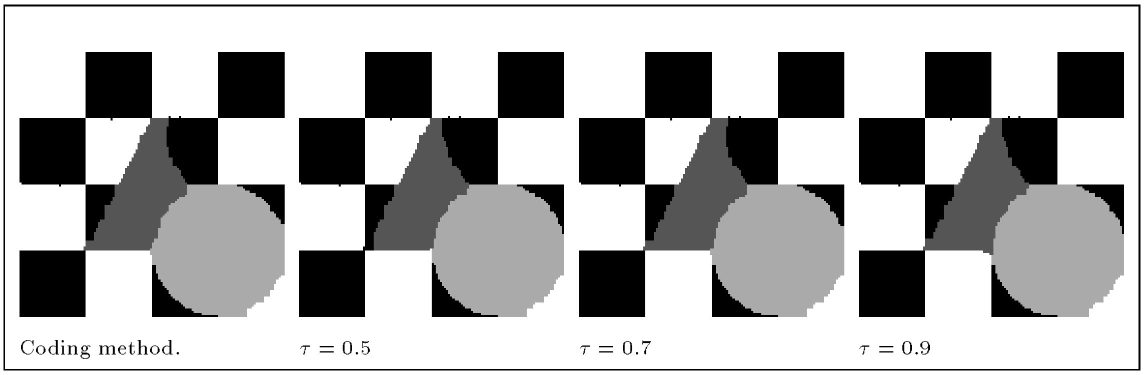

Figure 3.12: *Results of the Gibbs sampler with different data-parallel implementations.*

we have already explained how to implement a MTA relaxation on a hierarchical pyramid (cf. Figure 3.8). The parallelization of this scheme may be done by defining coding sets as described in Section 3.6.1. Since there are inter-level interactions, we cannot update each layer at the same time. In Figure 3.8, considering interactions between two neighboring grids, the levels connected by pointers are updated at the same time. Of course, at each level, we have to define additional coding sets corresponding to the intra-level communication scheme. For example, using a first order MRF model, we have to define 4 coding sets: 2 on the layers and 2 on the pyramid.

3.7 Experimental Results

The goal of this simulation is to evaluate the performances of the algorithms described in this chapter, in particular on image segmentation problems. The performances are evaluated in two ways: The reached global minimum of the energy function and the computer time required to attain this minimum. We remark that in all cases, the execution has been stoped when the energy change ΔU was less than 0.1% of the current value of U. In general, stochastic schemes are better regarding the achieved minimum and deterministic algorithms are better regarding the computer time. The methods have been tested on a variety of synthetic as well as real data using the monogrid segmentation model described in Section 2.3.

The MTA schedule has been compared to a classical inhomogeneous annealing using the hierarchical model described in Section 2.6.4.

All tests have been conducted on a Connection Machine CM200 [65] (see Sec-

Gibbs Sampler	VPR	Nb. of Iter	Total time	Time per It.	Energy
Coding	2	342	14.21 sec.	0.042 sec.	44190.63
$\tau = 0.5$	2	333	14.12 sec.	0.042 sec.	44195.77
$\tau = 0.7$	2	337	14.10 sec.	0.041 sec.	44190.28
$\tau = 0.9$	2	366	15.49 sec.	0.042 sec.	44192.86

Table 3.3: *Results of the Gibbs sampler with different data-parallel implementations.*

tion 2.7.1 for a discussion about it) using a coding method to implement parallelism (see Section 3.6.1 for more details). We have also tested the parallelization at rate τ described in Section 3.6.1 but we have obtained practically the same computing time and global energy as for the coding method. Since the convergence of the latter one is guaranteed to the same minima as for the sequential case, we decided to use it in the comparative tests. In Figure 3.12 we show the obtained results on a noisy synthetic image (the original image can be found in Figure 3.17). In Table 3.3, we give the computing time and the achieved energy minimum for each method.

3.7.1 Comparison of MTA and Inhomogeneous Annealing

In Figure 3.15, we compare the inhomogeneous and MTA schedules on a noisy synthetic image using the Gibbs sampler. The energy function of the hierarchical segmentation model is defined in Section 2.6.4. In both cases, the parameters were strictly the same, the only difference is the applied schedule: the pyramid contains 4 levels yielding a VPR equal to 4. The initial temperatures were respectively 4 (at the highest level), 3, 2 and 1 (at the lowest level) for MTA and 4 at each level for inhomogeneous annealing. The potential β equals to 0.7 and γ equals to 0.1. In Figure 3.14 (resp. 3.13), we show the global energy (computed at a fixed temperature) versus the number of iterations of the inhomogeneous (resp. MTA) schedule. Both reach practically the same minimum (53415.4 for the inhomogeneous and 53421.4 for the MTA), however the inhomogeneous schedule requires 238 iterations (796.8 sec. CPU time) but the MTA schedule requires only 100 iterations (340.6 sec. CPU time) for the convergence.

3.7.2 Stochastic and Deterministic Relaxation Algorithms

Herein, we present tests on a variety of images using the algorithms described in this chapter. For the simulations, we have used a first order MRF image model with the

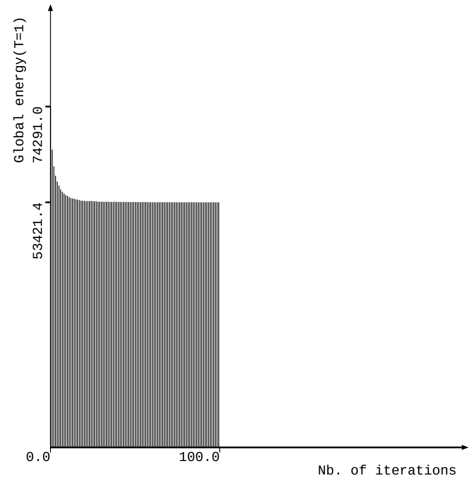

Figure 3.13: *Energy decrease with the MTA schedule.*

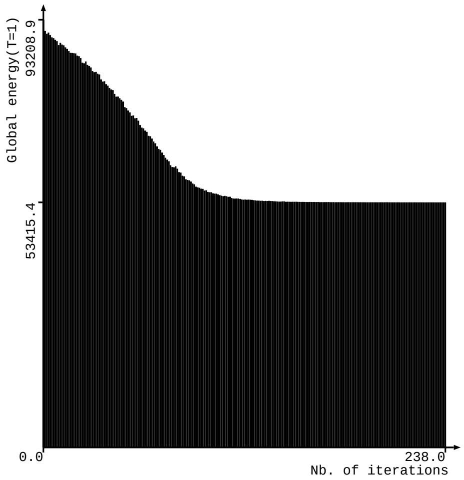

Figure 3.14: *Energy decrease with the inhomogeneous annealing schedule.*

following energy function (see Section 2.3 for more details):

$$U(\omega, f) = \sum_{s\in\mathcal{S}} \left(\ln(\sqrt{2\pi}\sigma_{\omega_s}) + \frac{(f_s - \mu_{\omega_s})^2}{2\sigma^2_{\omega_s}} \right) + \sum_{\{s,r\}\in\mathcal{C}} \beta\delta(\omega_s, \omega_r) \tag{3.77}$$

where

$$\delta(\omega_s, \omega_r) = \begin{cases} -1 & \text{if } \omega_s = \omega_r \\ +1 & \text{if } \omega_s \neq \omega_r \end{cases} \tag{3.78}$$

Image	α
checkerboard	0.3
triangle	0.3
bruit	0.7
SPOT	0.7

Table 3.4: *The α parameter for MMD and GSA.*

and β is a model parameter controlling the homogeneity of the regions. Each class $\lambda \in \Lambda$ is represented by its mean value μ_λ and its deviation σ_λ. $\omega_s \in \Lambda$ denotes the label attributed to the pixel $s \in \mathcal{S}$ and f_s stands for the grey-level value at pixel s. The model parameters are supposed to be known (cf. Table 3.5 in Appendix 3.D).

The initial temperature for the algorithms using annealing (that is Gibbs Sampler, Metropolis, MMD, GSA) was $T_0 = 4$ and the schedule is given by $T_{k+1} = 0.95 \cdot T_k$. For MMD and GSA, the parameter α has been chosen according to Table 3.4. For noisier synthetic images, we have chosen a smaller α to get the best result. For

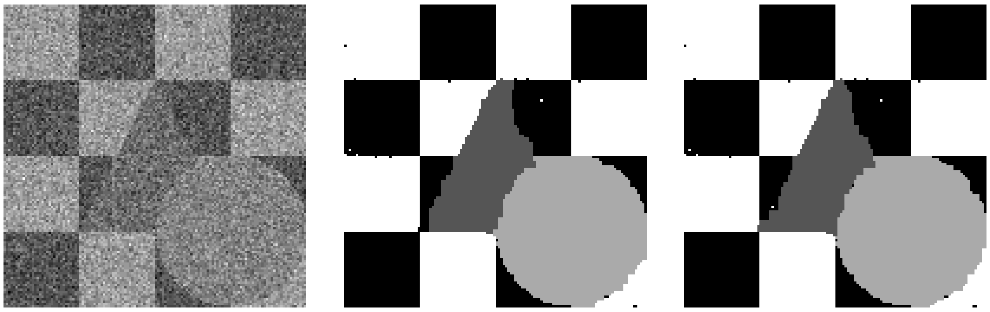

Figure 3.15: *Results of the Gibbs sampler on a synthetic image with inhomogeneous and MTA schedules.*

MMD, this means that we need a longer *stochastic phase* in order to get a better initialization for the *deterministic phase*. ICM and DPA was initialized by using only the Gaussian term of the energy function (for ICM, this means the *maximum likelihood* estimate of the labels). As for the other methods, *random* initial values were assigned to the labels. ICM is very sensitive to the initial conditions and may be better results could have been obtained with another initialization. Nevertheless the DPA and ICM algorithms have been initialized with the same data for the simulation.

The obtained results are presented in Appendix 3.C. In Appendix 3.D, we give for each image and each algorithm the Virtual Processor Ratio (VPR), the number of iterations, the computer time, and the reached energy minimum. As these results show, stochastic methods give the lowest energy values but they are slower than deterministic methods. ICM is the fastest but the reached minimum is much higher than for the other methods (as mentioned earlier, another initialization may lead to a better result, but more elaborated initialization usually increases the computer time). DPA, MMD and GSA seem to be a good compromise between quality and execution time. Sometimes, the results obtained by these algorithms are very close to the ones of stochastic methods. Another advantage is that they are far less dependent on the initialization than ICM.

Appendix

3.A Proof of The Multi-Temperature-Annealing Theorem

We follow the proof of the annealing theorem given by Geman and Geman in [46]. Essentially, we can apply the same proof, only a slight modification is needed.

3.A.1 Notations

We recall a few notations: $\mathcal{S} = \{s_1, s_2, \ldots, s_N\}$ denotes the set of sites, $\Lambda = \{0, 1, \ldots, L-1\}$ is a common state space and $\omega, \eta, \eta' \ldots \in \Omega$ denote configurations, where $\Omega = \Lambda^N$ is finite. The sites are updated in the order $\{n_1, n_2, \ldots\} \subset \mathcal{S}$. The generated configurations constitute an inhomogeneous Markov chain $\{X(k), k = 0, 1, 2, \ldots\}$, where $X(0)$ is the initial configuration. The transition $X(k-1) \to X(k)$ is controlled by the Gibbs distribution $\pi_{T(k,C)}$ according to the transition matrix at time k:

$$P_{\omega,\eta}(k) == \begin{cases} \pi_{T(k,C)}(X_{n_k} = \eta_{n_k} \mid X_s = \eta_s, s \neq n_k), & \text{if } \eta = \omega|_{\omega_{n_k}=\lambda} \text{ for some } \lambda \in \Lambda \\ 0, & \text{otherwise} \end{cases} \tag{3.79}$$

$\pi_{T(k,C)}(\omega)$ denotes the Gibbs distribution at iteration k

$$\pi_{T(k,C)}(\omega) = \frac{\exp(-U(\omega) \oslash T(k,C))}{Z} \tag{3.80}$$

$$\text{with } U(\omega) \oslash T(k,C) = \sum_{C \in \mathcal{C}} \frac{V_C(\omega)}{T(k,C)}\,. \tag{3.81}$$

The local characteristics of the above distribution are denoted by:

$$\pi_{T(k,C)}(X_s = \omega_s \mid X_r = \omega_r, s \neq r) = \frac{1}{Z_s} \exp\left(-\sum_{C \in \mathcal{C}: s \in C} \frac{V_C(\omega)}{T(k,C)}\right) \tag{3.82}$$

$$\text{with } Z_s = \sum_{\lambda \in \Lambda} \exp\left(-\sum_{C \in \mathcal{C}: s \in C} \frac{V_C(\omega|_{\omega_s=\lambda})}{T(k,C)}\right) \tag{3.83}$$

The decomposition of $U(\omega) - U(\eta)$ for arbitrary ω and η, $\omega \neq \eta$ is given by:

$$U(\omega) - U(\eta) = \sum_{C \in \mathcal{C}} (V_C(\omega) - V_C(\eta)). \tag{3.84}$$

Denoting respectively by $\Sigma^+(\omega, \eta)$ and $\Sigma^-(\omega, \eta)$ the sum over the positive and negative cliques, we get:

$$\sum_{C \in \mathcal{C}} (V_C(\omega) - V_C(\eta))$$

$$= \underbrace{\sum_{C\in\mathcal{C}:(V_C(\omega)-V_C(\eta))<0} (V_C(\omega) - V_C(\eta))}_{\Sigma^-(\omega,\eta)} + \underbrace{\sum_{C\in\mathcal{C}:(V_C(\omega)-V_C(\eta))\geq 0} (V_C(\omega) - V_C(\eta))}_{\Sigma^+(\omega,\eta)}. \quad (3.85)$$

Furthermore, let

$$U^{sup} = \max_{\omega\in\Omega} U(\omega), \quad (3.86)$$

$$U^{inf} = \min_{\omega\in\Omega} U(\omega), \quad (3.87)$$

$$\text{and } \Delta = U^{sup} - U^{inf}. \quad (3.88)$$

and define Σ_Δ^+ as the minimum of positive sums:

$$\Sigma_\Delta^+ = \min_{\substack{\omega' \in \Omega_{sup} \\ \omega'' \in \Omega_{opt}}} \Sigma^+(\omega', \omega''). \quad (3.89)$$

Obviously $\Delta \leq \Sigma_\Delta^+$.

Given any starting distribution μ_0, the distribution of $X(k)$ is given by the vector $\mu_0 \prod_{i=1}^k P(i)$:

$$P_{\mu_0}(X(k) = \omega) = \left(\mu_0 \prod_{i=1}^k P(i)\right)\Big|_\omega \quad (3.90)$$

$$= \sum_\eta P(X(k) = \omega | X(0) = \eta)\mu_0(\eta) \quad (3.91)$$

We use the following notation for transitions: $\forall l < k$ and $\omega, \eta \in \Omega$:

$$P(k, \omega | l, \eta) = P(X(k) = \omega | X(l) = \eta),$$

and for any distribution μ on Ω:

$$P(k, \omega | l, \mu) = \sum_\eta P(X(k) = \omega | X(l) = \eta)\mu(\eta).$$

Sometimes, we use this notation as $P(k, \cdot | l, \mu)$, where "$\cdot$" means *any* configuration from Ω. Finally, let $\|\mu - \nu\|$ denotes the following distance between two distributions on Ω:

$$\|\mu - \nu\| = \sum_\omega |\mu(\omega) - \nu(\omega)|.$$

It is clear, that $\lim_{n\to\infty} \mu_n = \mu$ in distribution (i.e. $\forall\omega : \mu_n(\omega) \to \mu(\omega)$) if and only if $\|\mu_n - \mu\| \to 0$.

3.A.2 Proof of the Theorem

First, we state two lemmas which imply Theorem 3.4.1:

Lemma 3.A.1 *For every* $k_0 = 0, 1, 2 \ldots$*:*

$$\lim_{k\to\infty} \sup_{\omega,\eta',\eta''} |P(X(k) = \omega | X(k_0) = \eta') - P(X(k) = \omega | X(k_0) = \eta'')| = 0. \qquad (3.92)$$

Proof of Lemma 3.A.1:
Fix $k_0 = 0, 1, 2, \ldots$, define $K_l = k_0 + l\kappa$, $l = 0, 1, 2, \ldots$, where κ is the number of transitions necessary for a full sweep of $\mathcal{S}$ (for every $k = 0, 1, 2, \ldots$: $\mathcal{S} \subseteq \{n_{k+1}, n_{k+2}, \ldots, n_{k+\kappa}\}$). Let $\delta(k)$ be the smallest probability among the local characteristics:

$$\delta(k) = \inf_{\substack{1\le i\le N\\ \omega\in\Omega}} \pi_{T(k,C)}(X_{s_i} = \omega_{s_i} | X_{s_j} = \omega_{s_j}, j \neq i).$$

A lower bound for $\delta(k)$ is given by:

$$\delta(k) \geq \frac{\exp(-U^{sup} \oslash T(k,C))}{L \exp(-U^{inf} \oslash T(k,C))} = \frac{\exp(-\Delta \oslash T(k,C))}{L} \geq \frac{1}{L} \exp(-\Sigma_\Delta^+ \oslash T(k,C)$$

$$\geq \frac{1}{L} \exp(-\Sigma_\Delta^+ / T_k^{inf}) \,,$$

where $L = | \Lambda |$ is the number of possible states at a site. Now fix l and define m_i as the time of the last replacement of site s_i before $K_l + 1$ (that is before the l^{th} full sweep):

$$\forall i\colon 1 \leq i \leq N : m_i = \sup\{k : k \leq K_l, n_k = s_i\}.$$

Without loss of generality, we can assume that $m_1 > m_2 \cdots > m_N$ (otherwise relabel the sites). Then:

$$P(X(K_l) = \omega | X(K_{l-1}) = \omega')$$

$$= P(X_{s_1}(m_1) = \omega_{s_1}, X_{s_2}(m_2) = \omega_{s_2}, \ldots, X_{s_N}(m_N) = \omega_{s_N} | X(K_{l-1}) = \omega')$$

$$= \prod_{i=1}^{N-1} P(X_{s_i}(m_i) = \omega_{s_i} | X_{s_{i+1}}(m_{i+1}) = \omega_{s_{i+1}}, \ldots, X_{s_N}(m_N) = \omega_{s_N}, X(K_{l-1}) = \omega')$$

$$\geq \prod_{i=1}^{N} \delta(m_i) \geq L^{-N} \prod_{i=1}^{N} \exp(-\Delta / T_{m_i}^{inf}) \geq L^{-N} \exp\left(-\frac{\Sigma_\Delta^+ N}{T_{k_0+l\kappa}^{inf}}\right) \qquad (3.93)$$

since $m_i \leq K_l = k_0 + l\kappa, i = 1, 2 \ldots, N$ and T_k^{inf} is decreasing. If $k_0 + l\kappa$ is sufficiently large then $T_{k_0+l\kappa}^{inf} \geq N\Sigma_\Delta^+ / \ln(k_0+l\kappa)$ according to condition 3.4.1/ii and Equation (3.93) can be continued as:

$$P(X(K_l) = \omega | X(K_{l-1}) = \omega') \geq L^{-N} \exp\left(-\frac{\Sigma_\Delta^+ N}{N\Sigma_\Delta^+ / \ln(k_0 + l\kappa)}\right) = L^{-N}(k_0 + l\kappa)^{-1}.$$

Hence, for a sufficiently small constant Γ $(0 < \Gamma \leq 1)$, we can assume that

$$\inf_{\omega,\omega'} P(X(K_l) = \omega | X(K_{l-1}) = \omega') \geq \frac{\Gamma L^{-N}}{k_0 + l\kappa} \tag{3.94}$$

for every $k_0 = 0, 1, 2, \ldots$ and $l = 1, 2, \ldots$, keeping in mind that K_l depends on k_0.

Consider now the limit given in Equation (3.92) and for each $k > k_0$, define $K^{sup}(k) = \sup\{l : K_l < k\}$ (the last sweep before the k^{th} transition) so that $\lim_{k\to\infty} K^{sup}(k) = \infty$. Fix $k > K_1$:

$$\sup_{\omega,\eta',\eta''} |P(X(k) = \omega | X(0) = \eta') - P(X(k) = \omega | X(0) = \eta'')|$$

$$= \sup_\omega \left(\sup_\eta P(X(k) = \omega | X(0) = \eta) - \inf_\eta P(X(k) = \omega | X(0) = \eta) \right)$$

$$= \sup_\omega \left(\sup_\eta \sum_{\omega'} P(X(k) = \omega | X(K_1) = \omega') P(X(K_1) = \omega' | X(0) = \eta) \right.$$

$$\left. - \inf_\eta \sum_{\omega'} P(X(k) = \omega | X(K_1) = \omega') P(X(K_1) = \omega' | X(0) = \eta) \right)$$

$$\doteq \sup_\omega Q(k, \omega).$$

Furthermore, for each $\omega \in \Omega$:

$$\sup_\eta \sum_{\omega'} P(X(k) = \omega | X(K_1) = \omega') P(X(K_1) = \omega' | X(0) = \eta)$$

$$\leq \sup_\mu \sum_{\omega'} P(X(k) = \omega | X(K_1) = \omega') \mu(\omega'),$$

where μ is any probability measure on Ω. Using Equation (3.94), we get:

$$\mu(\omega') \geq \frac{\Gamma L^{-N}}{k_0 + l\kappa}.$$

Suppose that $P(X(k)=\omega|X(K_1)=\omega')$ is maximized at $\omega'=\omega^{sup}$ and minimized at $\omega'=\omega^{inf}$. Then we get:

$$\sup_{\mu}\sum_{\omega'} P(X(k)=\omega|X(K_1)=\omega')\mu(\omega') \leq$$
$$\left(1-(L^N-1)\frac{\Gamma L^{-N}}{k_0+l\kappa}\right) P(X(k)=\omega|X(K_1)=\omega^{sup})$$
$$+\frac{\Gamma L^{-N}}{k_0+l\kappa} \underbrace{\sum_{\omega'\neq\omega^{sup}} P(X(k)=\omega|X(K_1)=\omega')}_{P(X(k)=\omega|X(K_1)=\omega^{inf})+\sum_{\omega'\neq\omega^{sup},\omega^{inf}} P(X(k)=\omega|X(K_1)=\omega')},$$

and in a similar way:

$$\inf_{\mu}\sum_{\omega'} P(X(k)=\omega|X(K_1)=\omega')\mu(\omega') \geq$$
$$\left(1-(L^N-1)\frac{\Gamma L^{-N}}{k_0+l\kappa}\right) P(X(k)=\omega|X(K_1)=\omega^{inf})$$
$$+\frac{\Gamma L^{-N}}{k_0+l\kappa} \underbrace{\sum_{\omega'\neq\omega^{inf}} P(X(k)=\omega|X(K_1)=\omega')}_{P(X(k)=\omega|X(K_1)=\omega^{sup})+\sum_{\omega'\neq\omega^{sup},\omega^{inf}} P(X(k)=\omega|X(K_1)=\omega')}.$$

Then, it is clear that

$$Q(k,\omega) \leq \left(1-\frac{\Gamma}{k_0+l\kappa}\right)\left(P(X(k)=\omega|X(K_1)=\omega^{sup}) - P(X(k)=\omega|X(K_1)=\omega^{inf})\right),$$

hence:

$$\sup_{\omega,\eta',\eta''} |P(X(k)=\omega|X(0)=\eta') - P(X(k)=\omega|X(0)=\eta'')| \leq$$
$$\left(1-\frac{\Gamma}{k_0+l\kappa}\right)\sup_{\omega,\eta',\eta''} |P(X(k)=\omega|X(K_1)=\eta') - P(X(k)=\omega|X(K_1)=\eta'')| \leq$$
$$\left(1-\frac{\Gamma}{k_0+l\kappa}\right)\left(\left(1-\frac{\Gamma}{k_0+l\kappa}\right)\sup_{\omega,\eta',\eta''} \left|P(X(k)=\omega|X(K_2)=\eta') - P(X(k)=\omega|X(K_2)=\eta'')\right|\right)$$

Proceeding this way, we have the following bound:

$$\leq \prod_{k=1}^{K^{sup}(k)} \left(1-\frac{\Gamma}{k_0+l\kappa}\right)\sup_{\omega,\eta',\eta''} \left|P(X(k)=\omega|X(K_{K^{sup}(k)})=\eta')\right.$$

$$-P(X(k) = \omega | X(K_{K^{sup}(k)}) = \eta'')\Big|$$

and finally, since the the possible maximal value of the supremum is 1:

$$\sup_{\omega,\eta',\eta''} |P(X(k) = \omega | X(0) = \eta') - P(X(k) = \omega | X(0) = \eta'')| \leq \prod_{k=1}^{K^{sup}(k)} \left(1 - \frac{\Gamma}{k_0 + l\kappa}\right).$$

It is then sufficient to show that

$$\lim_{m\to\infty} \prod_{k=1}^{m} \left(1 - \frac{\Gamma}{k_0 + l\kappa}\right) = 0.$$

which is a well known consequence of the divergence of the series

$$\sum_l (k_0 + l\kappa)^{-1}$$

for all k_0 and κ. This completes the proof of Lemma 3.A.1. **Q.E.D.**

Lemma 3.A.2

$$\lim_{k_0\to\infty} \sup_{k\geq k_0} \parallel P(k,\cdot|k_0,\pi_0) - \pi_0 \parallel = 0. \tag{3.95}$$

Proof of Lemma 3.A.2:

In the following, let $P_{k_0,k}(\cdot)$ stand for $P(k,\cdot|k_0,\pi_0)$, so that for any $k \geq k_0 > 0$:

$$P_{k_0,k}(\omega) = \sum_\eta P(X(k) = \omega | X(k_0) = \eta)\pi_0(\eta).$$

First, we show that for any $k > k_0 \geq 0$:

$$\|P_{k_0,k} - \pi_{T(k,C)}\| \leq \|P_{k_0,k-1} - \pi_{T(k,C)}\|. \tag{3.96}$$

We can assume for convenience that $n_k = s_1$. Then

$$\|P_{k_0,k} - \pi_{T(k,C)}\| =$$

$$\sum_{(\omega_{s_1},\ldots\omega_{s_N})} \Big|\pi_{T(k,C)}(X_{s_1} = \omega_{s_1} | X_s = \omega_s, s \neq s_1) P_{k_0,k-1}(X_s = \omega_s, s \neq s_1)$$

$$-\pi_{T(k,C)}(X_s = \omega_s, s \in \mathcal{S})\Big|$$

$$= \sum_{(\omega_{s_2},\ldots\omega_{s_N})} \left(\sum_{\omega_{s_1}\in\Lambda} \pi_{T(k,C)}(X_{s_1} = \omega_{s_1} | X_s = \omega_s, s \neq s_1) \, |P_{k_0,k-1}(X_s = \omega_s, s \neq s_1)\right.$$

$$-\pi_{T(k,C)}(X_s = \omega_s, s \neq s_1)\Big|\Big)$$

$$= \sum_{(\omega_{s_2},\ldots\omega_{s_N})} \Big|P_{k_0,k-1}(X_s = \omega_s, s \neq s_1) - \pi_{T(k,C)}(X_s = \omega_s, s \neq s_1)\Big|$$

$$= \sum_{(\omega_{s_2},\ldots\omega_{s_N})} \left|\sum_{\omega_{s_1}}(P_{k_0,k-1}(X_s = \omega_s, s \in \mathcal{S}) - \pi_{T(k,C)}(X_s = \omega_s, s \in \mathcal{S}))\right|$$

$$\leq \sum_{(\omega_{s_1},\ldots\omega_{s_N})} \Big|P_{k_0,k-1}(X_s = \omega_s, s \in \mathcal{S}) - \pi_{T(k,C)}(X_s = \omega_s, s \in \mathcal{S})\Big|$$

$$= \|P_{k_0,k-1} - \pi_{T(k,C)}\|.$$

Second, we prove that $\pi_{T(k,C)}$ converges to π_0 (the uniform distribution on Ω_{opt}):

$$\lim_{k\to\infty} \|\pi_0 - \pi_{T(k,C)}\| = 0.$$

To see this, let $|\Omega_{opt}|$ be the number of globally optimal configurations. Then

$$\lim_{k\to\infty} \pi_{T(k,C)}(\omega)$$

$$= \lim_{k\to\infty} \frac{\exp(-U(\omega) \oslash T(k,C))}{\sum_{\omega'\in\Omega_{opt}} \exp(-U(\omega') \oslash T(k,C)) + \sum_{\omega'\notin\Omega_{opt}} \exp(-U(\omega') \oslash T(k,C))}$$

$$= \lim_{k\to\infty} \frac{\exp(-(U(\omega) - U^{inf}) \oslash T(k,C))}{|\Omega_{opt}| + \sum_{\omega'\notin\Omega_{opt}} \exp(-(U(\omega) - U^{inf}) \oslash T(k,C))} = \begin{cases} 0 & \omega \notin \Omega_{opt} \\ \frac{1}{|\Omega_{opt}|} & \omega \in \Omega_{opt} \end{cases} \tag{3.97}$$

The above equation is true if $(U(\omega)-U^{inf})\oslash T(k,C) \geq 0$. Let us rewrite this inequality as

$$\sum_{C\in\mathcal{C}} \frac{V_C(\omega) - V_C(\omega')}{T(k,C)} \geq 0 \tag{3.98}$$

where ω' is any globally optimal configuration (i.e. $\omega' \in \Omega_{opt}$). While $V_C(\omega) - V_C(\omega')$ may be negative, $U(\omega) - U^{inf}$ is always positive or zero. We denote by $\Sigma(\omega)$ the energy difference in Equation (3.98) without the temperature. Obviously, it is non-negative:

$$\Sigma(\omega) = \sum_{C\in\mathcal{C}} V_C(\omega) - V_C(\omega') = U(\omega) - U^{inf} \geq 0$$

Then, let us decompose $\Sigma(\omega)$ according to Equation (3.55):

$$\Sigma(\omega) = \Sigma^+(\omega,\omega') + \Sigma^-(\omega,\omega').$$

From which:

$$\Sigma^{+}(\omega,\omega') = \Sigma(\omega) - \Sigma^{-}(\omega,\omega').$$

Now, we consider Equation (3.98):

$$\sum_{C\in\mathcal{C}} \frac{V_C(\omega) - V_C(\omega')}{T(k,C)} = \Sigma^{-}(\omega,\omega') \oslash T(k,C) + \Sigma^{+}(\omega,\omega') \oslash T(k,C)$$

$$\geq \Sigma^{-}(\omega,\omega')/T_k^{inf} + \Sigma^{+}(\omega,\omega')/T_k^{sup} = \frac{\Sigma^{-}(\omega,\omega')\cdot T_k^{sup} + \Sigma^{+}(\omega,\omega')\cdot T_k^{inf}}{T_k^{inf} T_k^{sup}} \geq 0$$

Furthermore:

$$\Sigma^{-}(\omega,\omega')\cdot T_k^{sup} + \Sigma^{+}(\omega,\omega')\cdot T_k^{inf} = \Sigma^{-}(\omega,\omega')\cdot T_k^{sup} + (\Sigma(\omega) - \Sigma^{-}(\omega,\omega'))T_k^{inf}$$

Therefore:

$$\Sigma^{-}(\omega,\omega')(T_k^{sup} - T_k^{inf}) - \Sigma(\omega)\cdot T_k^{inf} \geq 0$$

Dividing by $\Sigma^{-}(\omega,\omega')$ which is negative, we get:

$$T_k^{sup} - T_k^{inf} \leq \frac{\Sigma(\omega)}{\mid \Sigma^{-}(\omega,\omega') \mid} T_k^{inf}$$

Which is true due to condition 3.4.1/iii of the theorem.

Finally, we can prove that

$$\sum_{k=1}^{\infty} \left\| \pi_{T(k,C)} - \pi_{T(k+1,C)} \right\| < \infty \tag{3.99}$$

since

$$\sum_{k=1}^{\infty} \left\| \pi_{T(k,C)} - \pi_{T(k+1,C)} \right\| = \sum_{\omega} \sum_{k=1}^{\infty} \left| \pi_{T(k,C)}(\omega) - \pi_{T(k+1,C)}(\omega) \right|$$

and since

$$\forall \omega : \pi_{T(k,C)}(\omega) \longrightarrow \pi_0(\omega),$$

it is enough to show that $\pi_T(\omega)$ is monotonous for every ω. However it is clear from Equation (3.97) that

- if $\omega \notin \Omega_{opt}$ then $\pi_T(\omega)$ is strictly increasing for $0 < T \leq \epsilon$ for some sufficiently small ϵ,
- if $\omega \in \Omega_{opt}$ then $\pi_T(\omega)$ is strictly decreasing for all $T > 0$.

Fix $k > k_0 \geq 0$. From Equation (3.96) and Equation (3.99), we obtain:

$$\|P_{k_0,k} - \pi_0\| \leq \|P_{k_0,k} - \pi_{T(k,C)}\| + \|\pi_{T(k,C)} - \pi_0\|$$

$$\leq \|P_{k_0,k-1} - \pi_{T(k,C)}\| + \|\pi_{T(k,C)} - \pi_0\| \text{ by Equation (3.96)}$$

$$\leq \|P_{k_0,k-1} - \pi_{T(k-1,C)}\| + \|\pi_{T(k-1,C)} - \pi_{T(k,C)}\| + \|\pi_{T(k,C)} - \pi_0\|$$

$$\leq \|P_{k_0,k-2} - \pi_{T(k-2,C)}\| + \|\pi_{T(k-2,C)} - \pi_{T(k-1,C)}\| + \|\pi_{T(k-1,C)} - \pi_{T(k,C)}\| + \|\pi_{T(k,C)} - \pi_0\|$$

$$\leq \cdots \leq \|P_{k_0,k_0} - \pi_{T(k_0,C)}\| + \sum_{l=k_0}^{k-1} \|\pi_{T(l,C)} - \pi_{T(l+1,C)}\| + \|\pi_{T(k,C)} - \pi_0\|.$$

On the other hand,

$$P_{k_0,k_0} = \pi_0$$

and

$$\lim_{k\to\infty} \|\pi_{T(k,C)} - \pi_0\| = 0.$$

Then we have

$$\overline{\lim_{k_0\to\infty}} \sup_{k\geq k_0} \|P_{k_0,k} - \pi_0\| \leq \overline{\lim_{k_0\to\infty}} \sup_{k>k_0} \sum_{l=k_0}^{k-1} \|\pi_{T(l,C)} - \pi_{T(l+1,C)}\|$$

$$= \overline{\lim_{k_0\to\infty}} \sum_{l=k_0}^{\infty} \|\pi_{T(l,C)} - \pi_{T(l+1,C)}\| = 0$$

The last term is 0 by (3.99) which completes the proof of Lemma 3.A.1. **Q.E.D.**

Theorem 3.4.1 (Multi-Temperature Annealing) *Assume that there exists an integer $\kappa \geq N$ such that for every $k = 0, 1, 2, \ldots$, $\mathcal{S} \subseteq \{n_{k+1}, n_{k+2}, \ldots, n_{k+\kappa}\}$. For all $C \in \mathcal{C}$, let $T(k, C)$ be any decreasing sequence of temperatures in k for which*

(i) $\lim_{k\to\infty} T(k, C) = 0$.
Let us denote respectively by T_k^{inf} and T_k^{sup} the maximum and minimum of the temperature function at k ($\forall C \in \mathcal{C}$: $T_k^{inf} \leq T(k, C) \leq T_k^{sup}$).

(ii) For all $k \geq k_0$, for some integer $k_0 \geq 2$: $T_k^{inf} \geq N\Sigma_\Delta^+ / \ln(k)$.

(iii) If $\Sigma^-(\omega, \omega') \neq 0$ for some $\omega \in \Omega \setminus \Omega_{opt}$, $\omega' \in \Omega_{opt}$ then a further condition must be imposed:
For all k: $\frac{T_k^{sup} - T_k^{inf}}{T_k^{inf}} \leq R$ with

$$R = \min_{\substack{\omega \in \Omega \setminus \Omega_{opt} \\ \omega' \in \Omega_{opt} \\ \Sigma^-(\omega,\omega') \neq 0}} \frac{U(\omega) - U^{inf}}{\mid \Sigma^-(\omega, \omega') \mid}$$

Then for any starting configuration $\eta \in \Omega$ and for every $\omega \in \Omega$:

$$\lim_{k\to\infty} P(X(k) = \omega \mid X(0) = \eta) = \pi_0(\omega). \tag{3.100}$$

Proof:
Using the above mentioned lemmas, we can easily prove the annealing theorem:

$$\overline{\lim_{k\to\infty}} \|P(X(k) = \cdot|X(0) = \eta) - \pi_0\| = \overline{\lim_{k_0\to\infty} \lim_{\substack{k\to\infty \\ k\geq k_0}}} \|\sum_{\eta'} P(k, \cdot|k_0, \eta') P(k_0, \eta'|0, \eta) - \pi_0\|$$

$$\leq \overline{\lim_{k_0\to\infty} \lim_{\substack{k\to\infty \\ k\geq k_0}}} \|\sum_{\eta'} P(k, \cdot|k_0, \eta') P(k_0, \eta'|0, \eta) - P(k, \cdot|k_0, \pi_0)\|$$

$$+ \overline{\lim_{k_0\to\infty} \lim_{\substack{k\to\infty \\ k\geq k_0}}} \|P(k, \cdot|k_0, \pi_0) - \pi_0\| \ .$$

The last term is 0 by Lemma 3.A.2. Moreover, $P(k_0, \cdot|0, \eta)$ and π_0 have total mass 1, thus:

$$\|\sum_{\eta'} P(k, \cdot|k_0, \eta') P(k_0, \eta'|0, \eta) - P(k, \cdot|k_0, \pi_0)\|$$

$$= \sum_{\omega} \sup_{\eta''} |\sum_{\eta'} (P(k,\omega|k_0,\eta') - P(k,\omega|k_0,\eta''))(P(k_0,\eta'|0,\eta) - \pi_0(\eta'))|$$

$$\leq 2 \sum_{\omega} \sup_{\eta',\eta''} |P(k,\omega|k_0,\eta') - P(k,\omega|k_0,\eta'')| \ .$$

Finally,

$$\overline{\lim_{k\to\infty}} \|P(X(k) = \cdot|X(0) = \eta) - \pi_0\|$$

$$\leq 2 \sum_{\omega} \overline{\lim_{k_0\to\infty} \lim_{\substack{k\to\infty \\ k\geq k_0}}} \sup_{\eta',\eta''} |P(k,\omega|k_0,\eta') - P(k,\omega|k_0,\eta'')| = 0$$

The last term is 0 by Lemma 3.A.1 which completes the proof of the annealing theorem. **Q.E.D.**

3.B Proof of the MMD Theorem

3.B.1 Notations

Let Ω be the set of all configurations. The elements of Ω are denoted by $\omega, \eta, \ldots, U(\omega)$ denotes the energy of ω. A new configuration η is accepted according to the following rule:

$$\omega^{k+1} = \begin{cases} \eta & \text{if } \Delta U \leq 0, \\ \eta & \text{if } \Delta U > 0 \text{ and } \alpha \leq \exp\left(-\frac{\Delta U}{T}\right), \\ \omega^k & \text{otherwise} \end{cases} \tag{3.101}$$

where α is a constant threshold ($\alpha \in (0,1)$). For the convergence, we prove that after a certain number of iterations only transitions with lower energy are allowed.

3.B.2 Proof of the Theorem

Theorem 3.5.1 (MMD) *For any $\alpha \in (0,1)$, there exists a temperature threshold*

$$T_\alpha = -\frac{\Delta U_{min}}{\ln(\alpha)} \tag{3.102}$$

$$\textit{where } \Delta U_{min} = \min_{\substack{\omega, \eta \in \Omega \\ U(\omega) \neq U(\eta)}} \mid U(\omega) - U(\eta) \mid \tag{3.103}$$

such that if $T_k < T_\alpha$ then only configurations with lower energy will be accepted and thus the algorithm converges towards a local minimum.

Proof:
Let us examine the transition $\omega \rightarrow \eta$ when $U(\eta) > U(\omega)$. According to the acceptance rule in Equation (3.101), the jump is allowed if

$$\alpha \leq \exp\left(-\frac{U(\eta) - U(\omega)}{T}\right). \tag{3.104}$$

However, from Equation (3.103)

$$\alpha \leq \exp\left(-\frac{U(\eta) - U(\omega)}{T}\right) \leq \exp\left(-\frac{\Delta U_{min}}{T}\right), \tag{3.105}$$

On the other hand, $T \rightarrow 0$ as $k \rightarrow \infty$ yielding

$$\lim_{k \rightarrow \infty} \exp\left(-\frac{\Delta U_{min}}{T}\right) = 0. \tag{3.106}$$

Therefore, if $T < T_\alpha$, we get:

$$\exp\left(-\frac{\Delta U_{min}}{T}\right) \leq \exp\left(-\frac{\Delta U_{min}}{T_\alpha}\right) = \alpha. \tag{3.107}$$

Meaning that configurations with higher energy are no longer accepted. **Q.E.D.**

3.C Images

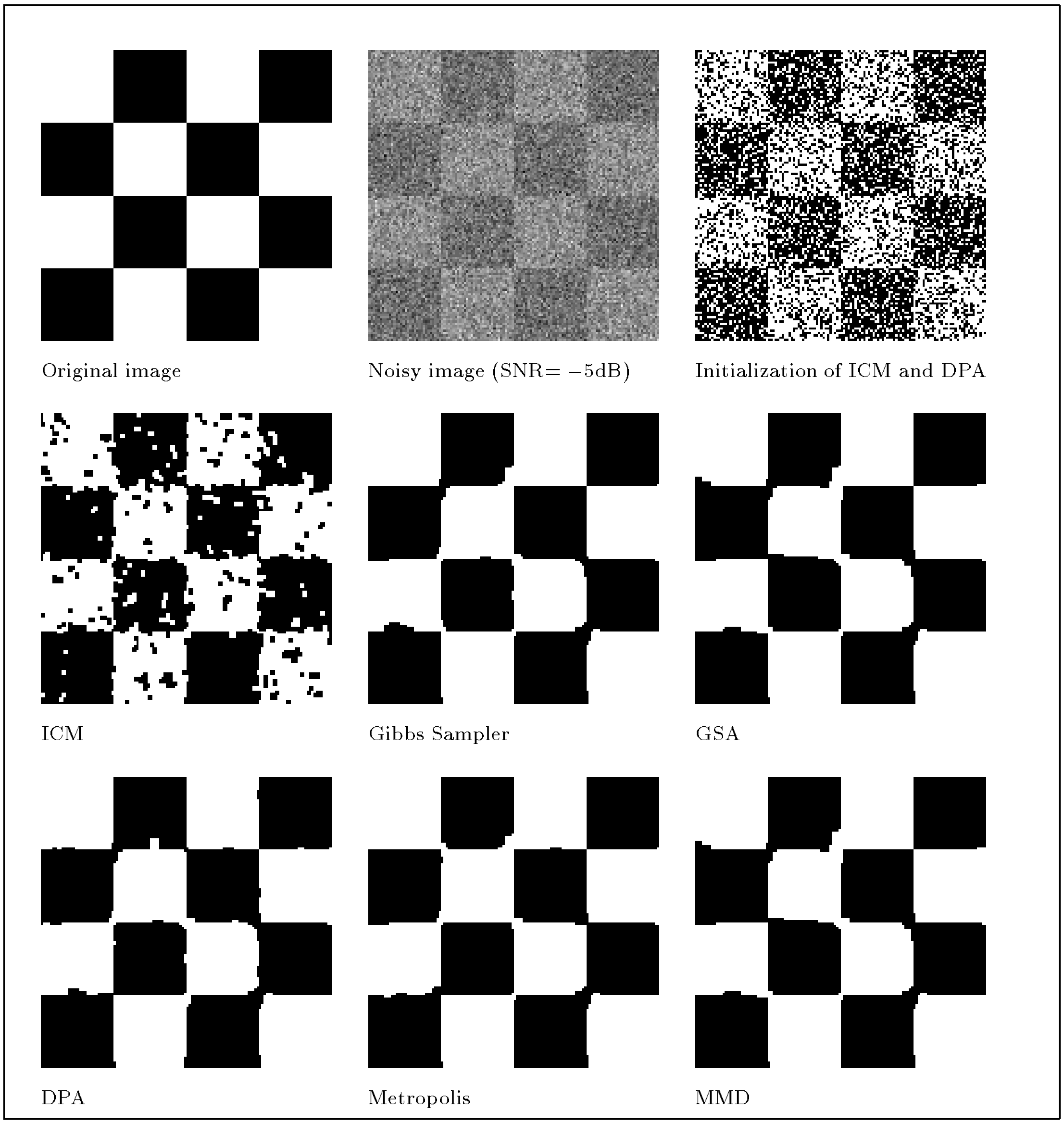

Figure 3.16: *Results on the "checkerboard" image with 2 classes.*

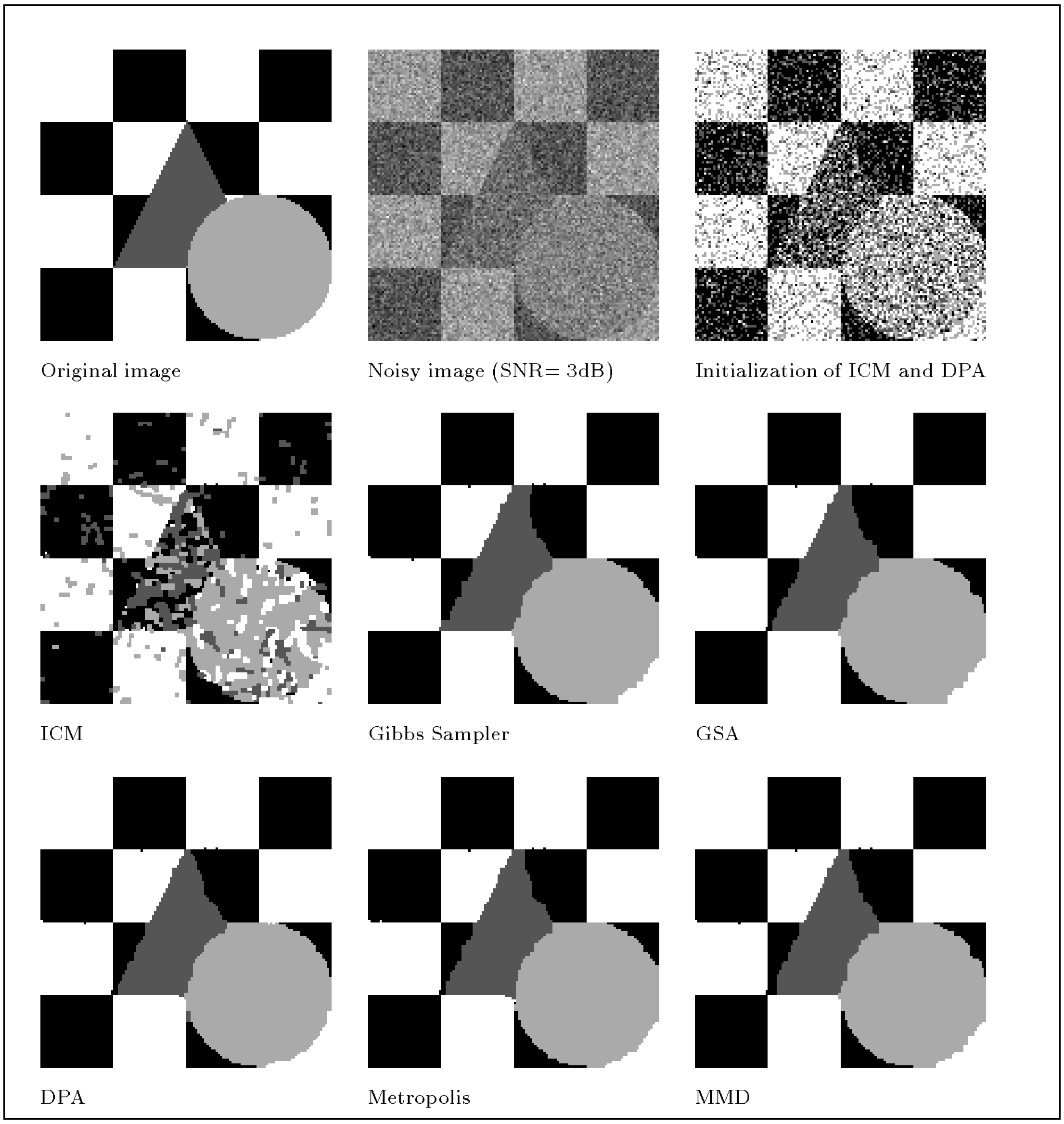

Figure 3.17: *Results on the "triangle" image with 4 classes.*

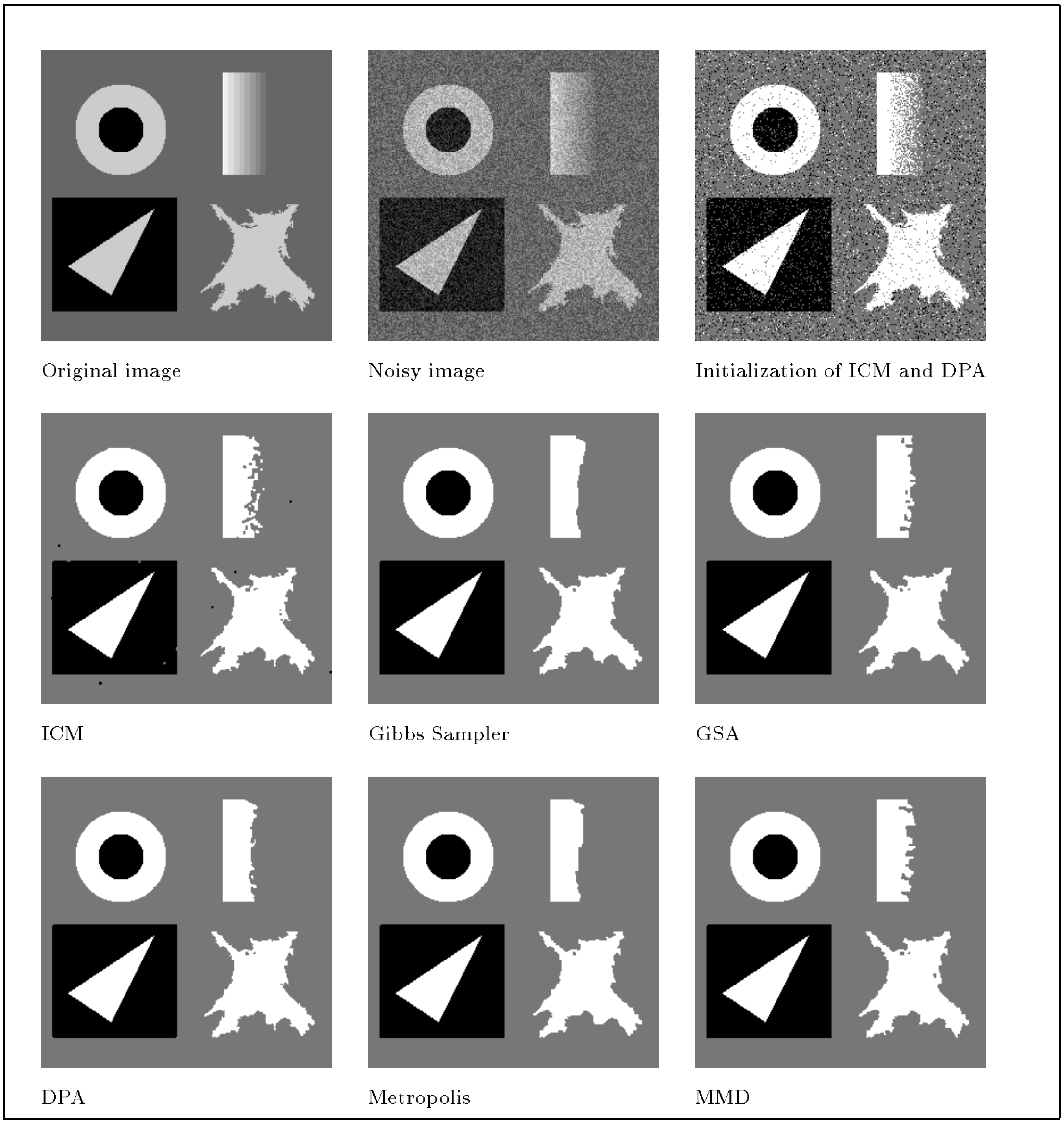

Figure 3.18: *Results on the "bruit" image with 3 classes.*

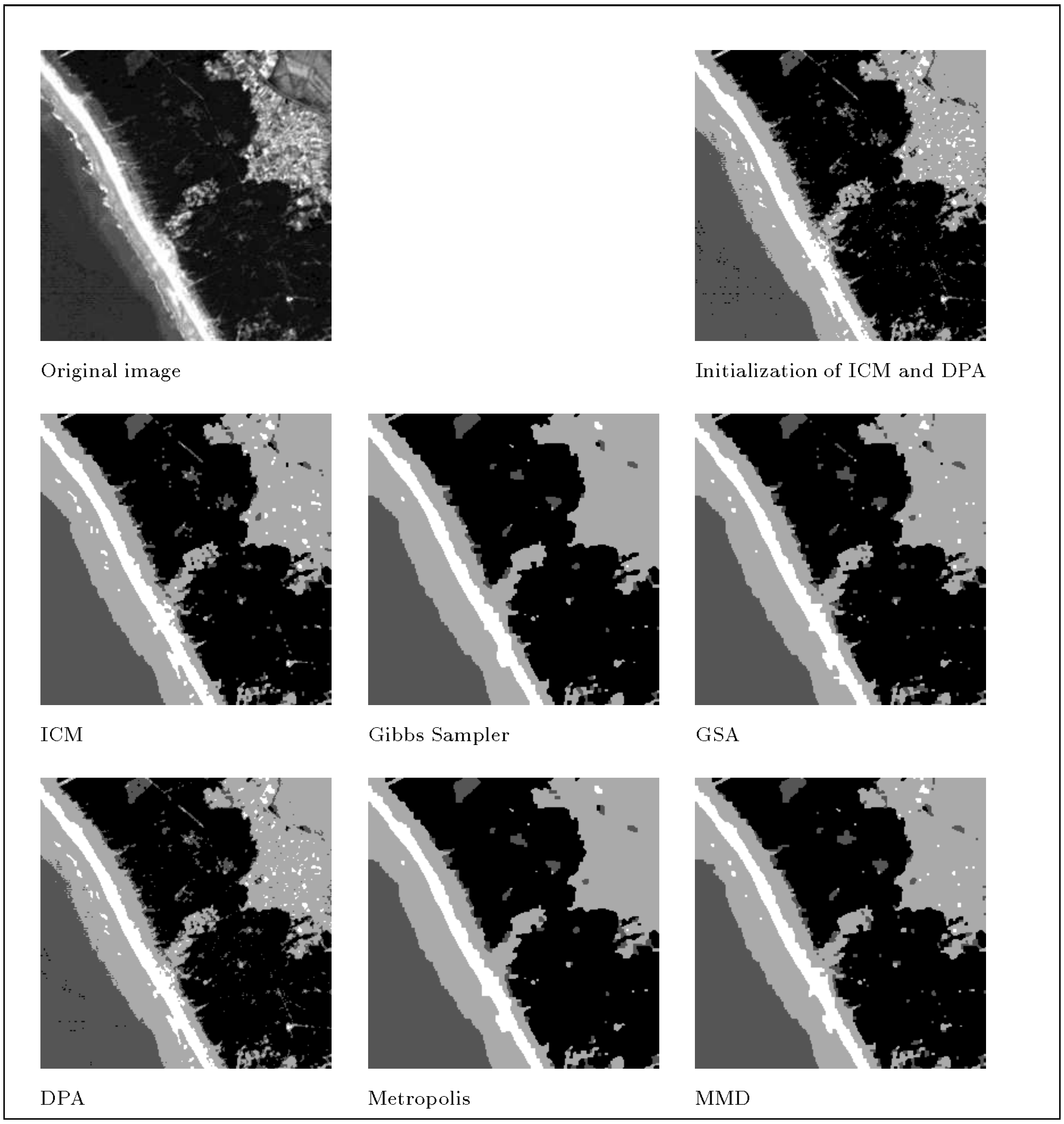

Figure 3.19: *Results on the "SPOT" image (4 classes).*

3.D Tables

Image	β	μ_1	σ_1^2	μ_2	σ_2^2	μ_3	σ_3^2	μ_4	σ_4^2
checkerboard	0.9	119.2	659.5	149.4	691.4	—	—	—	—
triangle	1.0	93.2	560.6	116.1	588.2	139.0	547.6	162.7	495.3
bruit	2.0	99.7	94.2	127.5	99.0	159.7	100.1	—	—
SPOT	2.0	30.3	8.2	37.4	4.6	61.3	128.1	98.2	127.1

Table 3.5: *Model parameters.*

	VPR	Nb. of Iter	Total time	Time per It.	Energy
ICM	2	8	0.078 sec.	0.009 sec.	52011.35
Metropolis	2	316	7.13 sec.	0.023 sec.	49447.60
Gibbs	2	322	9.38 sec.	0.029 sec.	49442.34
MMD	2	357	4.09 sec.	0.011 sec.	49459.60
GSA	2	357	7.59 sec.	0.021 sec.	49459.60
DPA	2	164	2.82 sec.	0.017 sec.	49458.02

Table 3.6: *Results on the "checkerboard" image with 2 classes.*

	VPR	Nb. of Iter	Total time	Time per It.	Energy
ICM	2	9	0.146 sec.	0.016 sec.	49209.07
Metropolis	2	202	7.31 sec.	0.036 sec.	44208.56
Gibbs	2	342	14.21 sec.	0.042 sec.	44190.63
MMD	2	292	7.41 sec.	0.025 sec.	44198.31
GSA	2	191	5.44 sec.	0.028 sec.	44198.88
DPA	2	34	1.13 sec.	0.033 sec.	44237.36

Table 3.7: *Results on the "triangle" image with 4 classes.*

	VPR	Nb. of Iter	Total time	Time per It.	Energy
ICM	2	8	0.302 sec.	0.037 sec.	-5552.06
Metropolis	2	287	37.33 sec.	0.130 sec.	-6896.59
Gibbs	2	301	35.76 sec.	0.118 sec.	-6903.68
MMD	2	118	10.15 sec.	0.086 sec.	-6216.50
GSA	2	242	17.84 sec.	0.073 sec.	-6256.00
DPA	8	15	1.33 sec.	0.089 sec.	-6685.52

Table 3.8: *Results on the "bruit" image with 3 classes.*

	VPR	Nb. of Iter	Total time	Time per It.	Energy
ICM	8	8	0.381 sec.	0.048 sec.	-52751.71
Metropolis	8	323	42.37 sec.	0.131 sec.	-58037.59
Gibbs	8	335	46.73 sec.	0.139 sec.	-58237.32
MMD	8	125	10.94 sec.	0.087 sec.	-56156.53
GSA	8	273	23.03 sec.	0.084 sec.	-56191.61
DPA	8	15	1.78 sec.	0.119 sec.	-40647.96

Table 3.9: *Results on the "SPOT" image with 4 classes.*

In This Chapter:

4. Parameter Estimation

In real life applications, the model parameters are usually unknown, one has to estimate [8] them only from the observable image. From a statistical viewpoint, this means that we want to estimate parameters from random variables whose joint distribution is a mixture of distributions. If we have a realization of the label field then the problem is relatively easy, we have many standard methods to do parameter estimation (Maximum Likelihood, Coding method [13], etc...). Unfortunately, such a realization is not known, so the direct use of such estimation algorithm is impossible. We have to approximate it by some function of the image data, which is the only observable attribute.

Some nowadays used algorithms are iterative [100, 90, 21], subsequently generating a labeling, estimating parameters from it, then generating a new labeling using these parameters, etc ...For such a method, we need a reasonably good initial value for each parameter. Since the classes of a labeling problem are mostly represented by a Gaussian distribution, the initialization of the mean and the variance of each class is very important because of its influence on subsequent labelings and hence on the final estimates. On the other hand, it is a classical problem, namely the determination of the modes of a Gaussian mixture without any a priori information. There are many approaches in this domain: Method of moments [40], Prony's Method [32] or geometrical analysis of the histogram [102], for instance.

Herein, we will discuss standard parameter estimation methods applied to monogrid and hierarchical MRF models. The presented algorithms have been tested on image segmentation problems. Comparative test have been done on noisy synthetic and real images.

4.1 The Parameter Estimation Problem

Let us briefly review some notations defined in Section 2.2 and Section 2.3. $\mathcal{F} = \{F_s : s \in \mathcal{S}\}$ denotes a set of image data on the sites (or pixels) $\mathcal{S} = \{s_1, s_2, \ldots, s_N\}$. Furthermore, each of these sites may take a label from $\Lambda = \{0, 1, \ldots, L-1\}$. The configuration space Ω is the set of all global discrete labelings $\omega = (\omega_{s_1}, \ldots, \omega_{s_N}), \omega_s \in \Lambda$. The label process is denoted by $\mathcal{X}$.

In parameter estimation problems, $\mathcal{F}$ is also called the *observed image* and $\mathcal{X}$ denotes the *unobserved* image attributes (labels). Furthermore, we are given n parameters forming a vector Θ which appears in the MRF model:

$$\Theta = \begin{pmatrix} \vartheta_1 \\ \vdots \\ \vartheta_n \end{pmatrix} \tag{4.1}$$

Up to now, Θ was considered to be known. Therefore, we were looking for the labeling which maximizes the a posteriori distribution

$$\widehat{\omega} = \arg\max_{\omega \in \Omega} P_\Theta(\omega \mid \mathcal{F}, \Theta). \tag{4.2}$$

where $\widehat{\omega}$ is the MAP estimate of the label field, given $\mathcal{F}$, under the model P_Θ (in the followings, the index Θ will be omitted). If both Θ and ω are unknown, the maximization problem in Equation (4.2) becomes [45, 84]

$$(\widehat{\omega}, \widehat{\Theta}) = \arg\max_{\omega, \Theta} P(\omega, \mathcal{F} \mid \Theta). \tag{4.3}$$

The pair $(\widehat{\omega}, \widehat{\Theta})$ is the global maximum of the joint probability $P(\omega, \mathcal{F} \mid \Theta)$. If we regard Θ as a random variable, the above maximization is an ordinary MAP estimation in the following way [45]: Let us suppose, that Θ is restricted to a finite volume domain $\mathcal{D}_\Theta$ and suppose that Θ is uniform on $\mathcal{D}_\Theta$ (that is $P(\Theta)$ is constant). Then, we get [45]:

$$\begin{aligned}
\arg\max_{\omega,\Theta} P(\omega, \Theta \mid \mathcal{F}) &= \arg\max_{\omega,\Theta} \frac{P(\omega, \mathcal{F} \mid \Theta) P(\Theta)}{P(\mathcal{F})} && (4.4)\\
&= \arg\max_{\omega,\Theta} \frac{P(\omega, \mathcal{F} \mid \Theta)}{\int_{\mathcal{D}_\Theta} \sum_{\omega \in \Omega} P(\omega, \mathcal{F} \mid \Theta) d\Theta} && (4.5)\\
&= \arg\max_{\omega,\Theta} P(\omega, \mathcal{F} \mid \Theta). && (4.6)
\end{aligned}$$

However, this maximization is very difficult, having no direct solution. Even SA is not implementable because the local characteristics with respect to the parameters Θ

cannot be computed from $P(\omega, \mathcal{F} \mid \Theta)$. One possible solution is to adopt the following criterion instead [45, 84]:

$$\widehat{\omega} = \arg\max_{\omega} P(\omega, \mathcal{F} \mid \widehat{\Theta}) \tag{4.7}$$

$$\widehat{\Theta} = \arg\max_{\Theta} P(\widehat{\omega}, \mathcal{F} \mid \Theta) \tag{4.8}$$

Clearly, Equation (4.7) is equivalent to Equation (4.3) for $\Theta = \widehat{\Theta}$ and Equation (4.8) is equivalent to Equation (4.3) with $\omega = \widehat{\omega}$. Furthermore, Equation (4.7) is equivalent to the MAP estimate of ω in the case of known parameters:

$$\arg\max_{\omega} P(\omega, \mathcal{F} \mid \widehat{\Theta}) = \arg\max_{\omega} P(\omega \mid \mathcal{F}, \widehat{\Theta}) P(\mathcal{F} \mid \widehat{\Theta}) = \arg\max_{\omega} P(\omega \mid \mathcal{F}, \widehat{\Theta}). \tag{4.9}$$

The solution of the MAP estimate has been discussed in details in the previous chapters.

4.2 Parameter Estimation from Fully Observed Data

The main study of this section is concentrated on Equation (4.8): the estimation of the parameters from a *labeled* sample. For a fixed labeling $\widehat{\omega}$, Equation (4.8) is equivalent to the so called *Maximum Likelihood* (ML) estimate of the parameters Θ based on the labeled sample $(\mathcal{F}, \widehat{\omega})$:

$$P(\widehat{\omega}, \mathcal{F} \mid \Theta) = P(\mathcal{F} \mid \widehat{\omega}, \Theta) P(\widehat{\omega} \mid \Theta) \tag{4.10}$$

4.2.1 Maximum Likelihood (ML)

The general form of a likelihood function is given by [8]:

$$L(\Theta) \equiv P(W \mid \Theta) \tag{4.11}$$

where W denotes the observations. In our case, $W = (\hat{\omega}, \mathcal{F})$. A strong argument for using ML estimate is its *consistency* and *asymptotic efficiency* (see [51] for a theoretical study). Another good reason is the generality and relative ease of application. Since $\ln(L(\Theta))$ is often a simpler function than $L(\Theta)$, we shall formulate the problem with the logarithmic likelihood function.

The likelihood function is an *objective function* which measures the departure of the data from the model. We seek the values $\widehat{\Theta}$ of the parameters for which the likelihood function attains its maximum. When the unknown parameters are free to assume any values, that is we have no a priori information about the parameters, then the above mentioned maximization problem is called *unconstrained optimization*. In the case of the ML estimate, we usually don't have a priori information thus we are concerned with such an optimization.

The following are *necessary* conditions for $\widehat{\Theta}$ to be a *maximum* of $\ln(L(\Theta))$ (and thus of $L(\Theta)$) [8]:

1. $\widehat{\Theta}$ is a *stationary* point of $\ln(L(\Theta))$:

$$\left.\frac{\partial \ln(L(\Theta))}{\partial \Theta}\right|_{\Theta=\widehat{\Theta}} = 0 \tag{4.12}$$

2. The Hessian matrix $H(\Theta)$ of $-\ln(L(\Theta))$ must be *positive semidefinite* for $\Theta = \widehat{\Theta}$, i.e., for any $\vec{y} \neq \vec{0}$:

$$\vec{y}^T H(\widehat{\Theta})\vec{y} \geq 0 \tag{4.13}$$

$$\text{where } H(\Theta) = \begin{pmatrix} \frac{\partial^2 -\ln(L(\Theta))}{\partial^2 \vartheta_1} & \frac{\partial^2 -\ln(L(\Theta))}{\partial \vartheta_1 \partial \vartheta_2} & \cdots & \frac{\partial^2 -\ln(L(\Theta))}{\partial \vartheta_1 \partial \vartheta_n} \\ \frac{\partial^2 -\ln(L(\Theta))}{\partial \vartheta_2 \partial \vartheta_1} & \frac{\partial^2 -\ln(L(\Theta))}{\partial^2 \vartheta_2} & \cdots & \frac{\partial^2 -\ln(L(\Theta))}{\partial \vartheta_2 \partial \vartheta_n} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial^2 -\ln(L(\Theta))}{\partial \vartheta_n \partial \vartheta_1} & \frac{\partial^2 -\ln(L(\Theta))}{\partial \vartheta_n \partial \vartheta_2} & \cdots & \frac{\partial^2 -\ln(L(\Theta))}{\partial^2 \vartheta_n} \end{pmatrix} \tag{4.14}$$

A *sufficient* condition for $\widehat{\Theta}$ to be a *local* maximum is that Equation (4.13) holds with strict inequality (i.e. $H(\widehat{\Theta})$ is *positive definite*). Moreover, if for all Θ, $H(\Theta)$ is positive definite and $\widehat{\Theta}$ is a stationary point then $\widehat{\Theta}$ is the *unique global maximum*.

Unfortunately, in MRF models, we cannot compute the likelihood function defined in Equation (4.10). While the first term is relatively easy to handle in independent Gaussian cases[1] [14, 12], the maximization of the second term is usually intractable due to the presence of the partition function $Z(\Theta)$:

$$P(\widehat{\omega} \mid \Theta) = \frac{\exp(-U(\widehat{\omega}, \Theta))}{Z(\Theta)} \tag{4.15}$$

$$\text{with } Z(\Theta) = \sum_{\omega \in \Omega} \exp(-U(\omega, \Theta)). \tag{4.16}$$

To tackle this problem, various approximations have been proposed [14, 13, 7, 25, 112]. We review herein two techniques: the coding method [14] and the Maximum Pseudo-Likelihood (MPL) [13, 25].

[1] The independent Gaussian case will be discussed later in Section 4.5.

4.2.2 Coding Method

A simple alternative to the ML estimate of $P(\widehat{\omega} \mid \Theta)$ is the coding method proposed by Besag [14]. We have already mentioned this method in an other context: in Section 3.6.1, we have used coding sets to implement data parallel optimization algorithms. The main purpose of the method in both cases is to build disjunct sets such that pixels belonging to the same set are conditionally independent. Indeed, if the neighborhood of a site $s \in \mathcal{S}$ is denoted by $\mathcal{V}_s$ then

$$s \in \mathcal{S}_i \Longleftrightarrow \forall r \in \mathcal{V}_s \colon r \notin \mathcal{S}_i \tag{4.17}$$

where $\mathcal{S}_i$ denotes a coding set. The number J of coding sets depends on the order of the neighborhood system used in the MRF model. In the case of a first order MRF model, we need two coding sets (cf. Figure 3.5). As a result of this decomposition, the maximum likelihood estimate of $P(\widehat{\omega} \mid \Theta)$ over a coding set $\mathcal{S}_i \subset \mathcal{S}$ becomes

$$P(\widehat{\omega} \mid \Theta) = \prod_{s \in \mathcal{S}_i} P(\widehat{\omega}_s \mid \widehat{\omega}_{\mathcal{V}_s}, \Theta), \tag{4.18}$$

which is a considerably easier task because the partition function is replaced by a *local* partition function Z_s defined by

$$Z_s = \sum_{\lambda \in \Lambda} \exp(-U(\widehat{\omega}|_{\widehat{\omega}_s = \lambda}, \Theta)), \tag{4.19}$$

which is easy to compute. Proceeding in this way, we obtain J estimates of Θ (one for each coding set), each of them is a reasonably good approximation of the ML estimate. The main drawback is that the method uses only part of the data and there is no optimal way to combine the J estimates. The most commonly used method is to take the average of the J approximations.

4.2.3 Maximum Pseudo-Likelihood (MPL)

A more efficient procedure is the Maximum Pseudo-Likelihood (MPL) [13, 25] estimator. It is simply the extension of Equation (4.18) over *all* sites in $\mathcal{S}$:

$$P(\widehat{\omega} \mid \Theta) = \prod_{s \in \mathcal{S}} P(\widehat{\omega}_s \mid \widehat{\omega}_{\mathcal{V}_s}, \Theta). \tag{4.20}$$

The original likelihood function is approximated by the product of local likelihoods. If the random variables $\omega_s, s \in \mathcal{S}$ are conditionally independent then the above equation is equivalent to the ML estimate, otherwise it is only an approximation. Many authors reported that the MPL approximation is more accurate than the coding method [84, 13, 49].

4.3 Parameter Estimation from Incomplete Data

In real life applications, labeled samples are usually not available. We have to estimate the parameters from an *unlabeled* sample. In statistics, the problem is known as the *incomplete data* problem. A broadly applicable algorithm has been proposed by Dempster *et al.* [31], called *Expectation – Maximization* (EM). The algorithm aims at determining the ML estimate of the parameters Θ by making use of the estimation of the missing data (i.e. the label field $\mathcal{X}$). Hereafter, we are also describing a few other estimation methods [26, 23, 45, 90, 100] available when dealing with incomplete data.

4.3.1 Expectation – Maximization (EM)

Let us denote the complete data by $W = (\mathcal{F}, \omega)$, which is the composite of the observed image $\mathcal{F}$ and the unobservable missing labels ω. The EM algorithm is an iterative procedure containing two steps: At iteration k, the E-step finds the expectation of the missing data $\widehat{\omega}^k$ given the parameters $\widehat{\Theta}^{k-1}$ obtained at the previous iteration. Then the M-step determines the ML estimate $\widehat{\Theta}^k$ of the parameters using the new estimate of the complete data, $\widehat{W}^k = (\mathcal{F}, \widehat{\omega}^k)$.

Algorithm 4.3.1 (EM)

① *Set* $k = 0$ *and initialize* $\widehat{\Theta}^0$.

② **(E-step)** *Estimate the complete data* $\widehat{W}^{k+1} = (\mathcal{F}, \widehat{\omega}^{k+1})$ *by finding*

$$\widehat{W}^{k+1} = E\{W \mid \mathcal{F}, \widehat{\Theta}^k\} \tag{4.21}$$

③ **(M-step)** *Compute* $\widehat{\Theta}^{k+1}$ *as the solution of the equations*

$$E\{W \mid \Theta\} = \widehat{W}^{k+1} \tag{4.22}$$

④ *Goto Step* ② *until* $\widehat{\Theta}$ *stabilizes.*

An application of the method to unsupervised image segmentation can be found in [86]. The novelty of the paper is to adopt the Mean Field approximation to compute estimates in the EM algorithm [115]. The main drawbacks of the EM algorithm are the large dependence on the initialization and also the computer time needed for the convergence.

4.3.2 Stochastic Expectation Maximization (SEM)

The SEM algorithm [23, 90] is an improvement of the EM algorithm by adding a stochastic step. SEM is less dependent on the initialization and convergence is faster than for the EM algorithm. The goal of the stochastic step is to generate a labeling $\widehat{\omega}^k$ according to the distribution $P(\omega \mid \mathcal{F}, \widehat{\Theta})$. In the E-step, we only compute the posterior distribution of the unobserved label field based on the current estimate of the parameters:

Algorithm 4.3.2 (SEM)

① *Set $k = 0$ and initialize $\widehat{\Theta}^0$.*

② **(E-step)** *Define the next distribution $P(\omega \mid \mathcal{F}, \widehat{\Theta}^k)$ based on the current parameter values $\widehat{\Theta}^k$.*

③ **(S-step)** *Sample from the label field $\mathcal{X}$ according to the distribution $P(\omega \mid \mathcal{F}, \widehat{\Theta}^k)$ and denote the resulting labeling by $\widehat{\omega}^{k+1}$.*

④ **(M-step)** *The same as for the EM algorithm: Compute $\widehat{\Theta}^{k+1}$ as the solution of the equations*

$$E\{W \mid \Theta\} = \widehat{W}^{k+1} \tag{4.23}$$

with $\widehat{W}^{k+1} = (\mathcal{F}, \widehat{\omega}^{k+1})$.

⑤ *Goto Step ② with $k = k + 1$ until $\widehat{\Theta}$ stabilizes.*

While EM is a deterministic algorithm, SEM is a stochastic algorithm using a stochastic sampling instead of the fully deterministic E-step of EM. For SEM, the E-step only defines the next distribution which controls the sampling in the S-step. As claimed in [90, 21], the main advantage of the SEM algorithm compared to the classical EM is to avoid the local maxima of the likelihood function, resulting in a better estimate of the parameters.

4.3.3 Adaptive Simulated Annealing (ASA)

Another EM-like algorithm has been proposed by Geman in [45], whis is called *Adaptive Simulated Annealing* (ASA). The algorithm was adapted to image segmentation problems in [84], where the convergence of ASA has also been proved. The ASA algorithm is very similar to the SEM, it may be seen as a special case where the S-step is implemented by a Simulated Annealing.

Algorithm 4.3.3 (ASA)

① *Set* $k = 0$ *and initialize* $\widehat{\Theta}^0$.

② *Do* n *iterations* ($n \geq 1$) *of SA sampling from* $P(\omega \mid \mathcal{F}, \widehat{\Theta}^k)$. *The resulting labeling is denoted by* $\widehat{\omega}^{k+1}$.

③ *Update the current estimate of the parameters,* $\widehat{\Theta}^{k+1}$ *to the ML estimate based on the current labeling* $\widehat{\omega}^{k+1}$.

④ *Goto Step* ② *with* $k = k + 1$ *until* $\widehat{\Theta}$ *stabilizes.*

If ML estimate is not tractable, which is often the case when dealing with MRF models, one can use an approximation (MPL, for instance). We remark that a similar algorithm has been reported in [13]. It uses ICM instead of SA in Step ②.

4.3.4 Iterative Conditional Estimation (ICE)

Finally, we present a solution to the incomplete data problem proposed by Pieczynski *et al.* [100, 19, 1]. Let us consider an estimator $\mathcal{E}_\Theta(\mathcal{F}, \omega)$ of Θ (ML, for instance). Since the current state of the label field is unknown, the direct use of $\mathcal{E}_\Theta(\mathcal{F}, \omega)$ is impossible, we have to approximate it. The best approximation, in the mean-square sense, is the *conditional expectation*. Since $E\{\mathcal{E}_\Theta \mid \mathcal{F}, \omega\}$ depends on the parameters Θ, we need a parameter $\widehat{\Theta}^k$ previously defined by some way. This defines an iterative procedure, called ICE [100, 19, 101]:

Algorithm 4.3.4 (ICE)

① *Set* $k = 0$ *and initialize* $\widehat{\Theta}^0$.

② *Generate* n *realizations* (n *is fixed a priori*) $\widehat{\omega}^i (1 \leq i \leq n)$ *of the label field based on* $\widehat{\Theta}^k$.

③ *Based on the sample* $\widehat{\omega}^i (1 \leq i \leq n)$, $\widehat{\Theta}^{k+1}$ *is obtained as the conditional expectation*

$$\widehat{\Theta}^{k+1} = E\{\mathcal{E}_\Theta \mid \mathcal{X} = \omega\} \approx \frac{1}{n} \sum_{i=1}^{n} \mathcal{E}_\Theta(\mathcal{F}, \widehat{\omega}^i). \tag{4.24}$$

④ *Goto Step* ② *until* $\widehat{\Theta}$ *stabilizes.*

Compared to the EM algorithm, ICE results in a better estimate of the parameters and the convergence is faster [19, 1].

4.4 Gaussian Mixture Identification

In image labeling problems, the classes are often modeled by a Gaussian distribution. What we observe is a *mixture* of Gaussian distributions and we have to determine the modes of this mixture corresponding to the classes. The problem is to estimate the parameters of these modes from an *unlabeled* sample. The methods, presented in the previous section (EM, SEM, etc...), can also be adapted to this problem. Herein, we are interested in non-iterative methods.

4.4.1 Geometrical Identification

In [102], a geometrical analysis of the mixture density function is used: First, let us consider a n-dimensional normal density function $f(\vec{x})$ (defined in Equation (1.46)) and review the relationships between the geometrical characteristics of the concave domain of $f(\vec{x})$ and its mean vector $\vec{\mu}$ and covariance matrix Σ (see Section 1.4 for more details about normal distributions). Considering a zero-mean density,

$$f(\vec{x}) = \frac{1}{\sqrt{(2\pi)^n|\Sigma|}} \exp\left(-\frac{1}{2}\vec{x}^T\Sigma^{-1}\vec{x}\right). \tag{4.25}$$

Let us denote by Q the following quadratic form:

$$Q = \vec{x}^T\Sigma^{-1}\vec{x} \tag{4.26}$$

Using a change of basis $\vec{x} = M\vec{y}$, where M is an $n \times n$ orthogonal transformation matrix to be precised later, we get

$$Q = \vec{y}^T M^T \Sigma^{-1} M\vec{y}. \tag{4.27}$$

$M^T\Sigma^{-1}M$ can be further simplified: Since Σ^{-1} is real and symmetric, it can always be diagonalized by an appropriate orthogonal transformation. Choosing M as the eigenvectors of the matrix Σ^{-1} yields the following form:

$$Q = \vec{y}^T\Lambda\vec{y} \qquad \text{with } \Lambda = \begin{pmatrix} \lambda_1 & & & \\ & \lambda_2 & & \mathbf{0} \\ & & \ddots & \\ \mathbf{0} & & & \lambda_n \end{pmatrix}, \tag{4.28}$$

where Λ is a diagonal matrix with eigenvalues in its diagonal. After this transformation, Equation (4.25) is of the following form:

$$f(\vec{x}) = \frac{1}{\sqrt{(2\pi)^n |\Sigma|}} \exp\left(-\frac{1}{2}\vec{y}^T \Lambda \vec{y}\right). \tag{4.29}$$

In [102], it is shown that the domain in which $f(\vec{x})$ is concave is the interior of the quadratic surface defined by

$$\vec{y}^T \Lambda \vec{y} = 1. \tag{4.30}$$

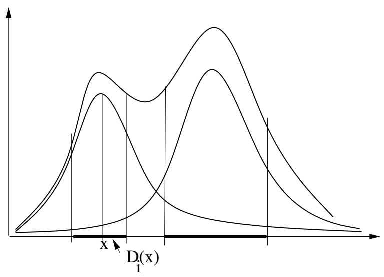

Figure 4.1: *Concave domains of an univariate Gaussian mixture.*

This is a hyperellipsoid, as already discussed in Section 1.4 (see Figure 1.3 and Equation (1.50)). Its center is the mean vector $\vec{\mu}$ and its principal axes coincide with the eigenvectors of Σ^{-1}. Furthermore, there is a relationship between the length ℓ_i of each principal axis and the corresponding eigenvalue λ_i:

$$\ell_i = \frac{2}{\sqrt{\lambda_i}} \tag{4.31}$$

A concave domain of $f(\vec{x})$ is identified in the following way: To each point $\vec{x}$ is associated a family of domains centered at $\vec{x}$ (for example a sequence of larger and larger hypercubes). They are denoted by $D_i(\vec{x})$. The mean value of $f(\vec{x})$ within $D_i(\vec{x})$ is defined as:

$$E\{D_i(\vec{x})\} = \frac{\int_{D_i(\vec{x})} f(\xi) d\xi}{\int_{D_i(\vec{x})} d\xi}. \tag{4.32}$$

The test of convexity is based on the fact that $E\{D_i(\vec{x})\}$ is monotonic decreasing function of i for any family of domains $D_i(\vec{x})$ standing in a concave region of $f(\vec{x})$.

The above described technique is used to determine the modes (that is the concave domains) of the image histogram (regarded as a mixture of normal distributions) and then to compute their parameters. Now, we examine these equations in the case of an univariate mixture ($n = 1$) which is the most interesting for image segmentation problems. Let us denote the histogram values of an image by $h_0, h_1, \ldots, h_G$ ($h_x \in [0, 1]$), where G is the number of possible grey-levels. The domains $D_i(x)$ are simply line segments centered at x (see Figure 4.1). The computer algorithm, which computes the modes of the histogram, is the following:

Algorithm 4.4.1 (Gaussian Mixture Identification)

① *Define a small neighborhood* D_1 *and a larger one* D_2 *at each point* $x \in [0, G]$. *Compute* $E\{D_1(x)\}$ *and* $E\{D_2(x)\}$ *by approximating Equation (4.32) through*

$$E\{D_i(x)\} \approx \frac{\sum_{\xi \in D_i(x)} h_\xi}{l(D_i(x))} \tag{4.33}$$

where $l(D_i(x))$ *is the length of the line segment* D_i. *Compute* $\delta(x) = E\{D_2(x)\} - E\{D_1(x)\}$ *at each* x. *If* $\delta(x) \leq 0$ *then the histogram is concave in the corresponding domain* $D_1(x)$.

② *All adjacent domains, where the histogram is found concave, have to be aggregated. The so-obtained domains* $\widehat{D}_k (k = 1, \ldots, L)$ *characterize the modes of the mixture and* L *is the number of modes or classes.*

③ *The mean* μ_k *of mode* k *is the center of the domain* $\widehat{D}_k$ *and* σ_k *equals to the length of* $\widehat{D}_k$. *Representing the concave domains* $\widehat{D}_k$ *by* (x_1^k, x_2^k), *we get:*

$$\mu_k = \frac{x_2^k - x_1^k}{2} \quad \text{and } \sigma_k = x_2^k - x_1^k. \tag{4.34}$$

In Step ②, we obtain not only the concave domains but also the number of modes corresponding to the number of classes. Usually, we obtain too many classes since the convexity test in Step ① is too sensitive to local irregularities. Consequently, we have to eliminate some classes, in particular classes with small a priori probability which can be approximated by

$$P_k \propto \sum_{x \in \widehat{D}_k} h_x \tag{4.35}$$

Another method would be to a priori fix the number of classes L and then to keep the L most likely domains (we have used this technique in the simulations).

4.4.2 Method of Moments

Another non geometrical method is the *method of moments* [40]. This approach is based on equating the sample moments computed from the data to the corresponding mixture moments. Let us suppose that the mixture consists of L univariate normal distributions with parameters μ_i and σ_i. The mixture density is given by

$$f(x) = \sum_{i=1}^{L} P_i f_i(x) \tag{4.36}$$

where $f_i(x)$ is the i^{th} normal density function and $P_i(1 \leq i \leq L)$ is the weight or *a priori probability* of the i^{th} component. The *method of moments* consists in generating equations

$$\begin{aligned} \widehat{m}_k &= m_k(P_1, \ldots, P_L; \mu_1, \ldots, \mu_L; \sigma_1, \ldots, \sigma_L) && (4.37)\\ &= \sum_{i=1}^{L} P_i m_k^i(\mu_i, \sigma_i) && (4.38) \end{aligned}$$

where $\widehat{m}_k$ is the k^{th} sample moment, m_k is the mixture moment and m_k^i is the moment of the i^{th} Gaussian component. As we already explained in Section 1.4, the moments of a Gaussian random variable can be expressed by a recursive formula in terms of its mean value μ and its deviation σ (see Equation (1.52) – Equation (1.56)). For each moment, Equation (4.38) yields a nonlinear equation which has to be solved in order to get the estimates of the unknown parameters. The solution of these equations [40] is so complicated that it is practically unusable for a more than two component case.

If the component distributions have equal means or equal variances, the nonlinear moment equations can be transformed into a set of linear equations using Prony's method [32]. The solution is then feasible for more than two components.

4.5 Unsupervised Image Segmentation

Herein, we consider the monogrid MRF segmentation model presented in Section 2.3 but with unknown parameters [77]. Let us first review the model. We are given the grey-levels $\mathcal{F}$ of an image $\mathcal{S} = \{s_1, s_2, \ldots, s_N\}$, which is the only observable attribute. Moreover, we are given a set of labels denoted by $\Lambda = \{0, 1, \ldots, L-1\}$. The problem is to estimate the model parameters Θ and find the MAP estimate of the label field $\mathcal{X}$ among the possible discrete labelings $\Omega = \Lambda^N = \{\omega = (\omega_{s_1}, \ldots, \omega_{s_N}) : \omega_s \in \Lambda\}$. As explained in Section 4.2, in the case of unknown parameters, the maximization problem becomes (cf. Equation (4.3)):

$$(\widehat{\omega}, \widehat{\Theta}) = \arg\max_{\omega, \Theta} P(\omega, \mathcal{F} \mid \Theta). \tag{4.39}$$

Since this maximization is not tractable, we use instead Equation (4.7) and Equation (4.8). For the maximization problem in Equation (4.7), which is the ordinary

MAP estimate with known parameters, we refer to Section 2.3. Herein, we are interested in the solution of the ML estimation using Equation (4.8):

$$\widehat{\Theta} = \arg\max_{\Theta} P(\widehat{\omega}, \mathcal{F} \mid \Theta) \tag{4.40}$$

According to Equation (4.10), the probability at the right hand side can be written as

$$P(\widehat{\omega}, \mathcal{F} \mid \Theta) = P(\mathcal{F} \mid \widehat{\omega}, \Theta) P(\widehat{\omega} \mid \Theta) \tag{4.41}$$

Using the model defined in Section 2.3, the first term is a product of independent Gaussian densities and the second term is a first order MRF, also known as the Potts model in statistical mechanics:

$$\begin{aligned} P(\widehat{\omega}, \mathcal{F} \mid \Theta) &= \prod_{s \in \mathcal{S}} \frac{1}{\sqrt{2\pi}\sigma_{\widehat{\omega}_s}} \exp\left(-\frac{(f_s - \mu_{\widehat{\omega}_s})^2}{2\sigma^2_{\widehat{\omega}_s}}\right) \\ &\quad \cdot \frac{\exp(-2\beta \sum_{\{s,r\} \in \mathcal{C}} \delta(\widehat{\omega}_s, \widehat{\omega}_r))}{Z(\beta)} \end{aligned} \tag{4.42}$$

$$\text{with } Z(\beta) = \sum_{\omega \in \Omega} \exp\left(-2\beta \sum_{\{s,r\} \in \mathcal{C}} \delta(\omega_s, \omega_r)\right) \tag{4.43}$$

$$\text{and } \delta(\widehat{\omega}_s, \widehat{\omega}_r) = \begin{cases} 0 & \text{if } \widehat{\omega}_s = \widehat{\omega}_r \\ 1 & \text{otherwise} \end{cases} \tag{4.44}$$

We have $2L + 1$ parameters (two for each class and one hyperparameter β):

$$\Theta = \begin{pmatrix} \mu_0 \\ \mu_1 \\ \vdots \\ \mu_{L-1} \\ \sigma_0 \\ \vdots \\ \sigma_{L-1} \\ \beta \end{pmatrix} \tag{4.45}$$

The first $2L$ parameters are estimated from the Gaussian term and the last one is computed from the Markovian term. Instead of the likelihood function defined in Equation (4.42), we consider the simpler logarithmic likelihood:

$$\ln(L(\Theta)) = \sum_{s \in \mathcal{S}} \left(-\ln(\sqrt{2\pi}\sigma_{\widehat{\omega}_s}) - \frac{(f_s - \mu_{\widehat{\omega}_s})^2}{2\sigma^2_{\widehat{\omega}_s}}\right)$$

$$
\begin{aligned}
& -2\beta \sum_{\{s,r\}\in\mathcal{C}} \delta(\widehat{\omega}_s, \widehat{\omega}_r) - \ln(Z(\beta)) & (4.46) \\
= & \underbrace{\sum_{\lambda\in\Lambda} \sum_{s\in\mathcal{S}_\lambda} \left(-\ln(\sqrt{2\pi}\sigma_\lambda) - \frac{(f_s-\mu_\lambda)^2}{2\sigma_\lambda^2} \right)}_{\mathcal{G}(\mu_\lambda,\sigma_\lambda)} \\
& \underbrace{-2\beta \sum_{\{s,r\}\in\mathcal{C}} \delta(\widehat{\omega}_s, \widehat{\omega}_r) - \ln(Z(\beta))}_{\mathcal{M}(\beta)} & (4.47)
\end{aligned}
$$

where $\mathcal{S}_\lambda$ is the set of pixels where $\widehat{\omega} = \lambda$. Using the results reported in Section 4.2.1, the estimate $\widehat{\Theta}$ must satisfy Equation (4.12) and Equation (4.13). From Equation (4.12), we obtain the following equations:

$$
\begin{aligned}
\forall\lambda\in\Lambda: \quad \frac{\partial\mathcal{G}(\mu_\lambda,\sigma_\lambda)}{\partial\mu_\lambda} &= 0 & (4.48) \\
\frac{\partial\mathcal{G}(\mu_\lambda,\sigma_\lambda)}{\partial\sigma_\lambda} &= 0 & (4.49) \\
\text{and } \frac{\partial\mathcal{M}(\beta)}{\partial\beta} &= 0 & (4.50)
\end{aligned}
$$

The solution of the above system for μ_λ and σ_λ is simply the empirical mean and variance:

$$
\begin{aligned}
\forall\lambda\in\Lambda: \quad \mu_\lambda &= \frac{1}{\mid\mathcal{S}_\lambda\mid} \sum_{s\in\mathcal{S}_\lambda} f_s, \\
\sigma_\lambda^2 &= \frac{1}{\mid\mathcal{S}_\lambda\mid} \sum_{s\in\mathcal{S}_\lambda} (f_s-\mu_\lambda)^2. & (4.51)
\end{aligned}
$$

The solution for β, however, is not as easy. Let us consider the derivative of $\mathcal{M}(\beta)$:

$$
\begin{aligned}
& \frac{\partial}{\partial\beta}\left(-2\beta N^{ih}(\widehat{\omega}) - \ln\left(\sum_{\omega\in\Omega} \exp(-2\beta N^{ih}(\omega)) \right) \right) \\
& = -N^{ih}(\widehat{\omega}) + \frac{\sum_{\omega\in\Omega} N^{ih}(\omega)\exp(-2\beta N^{ih}(\omega))}{\sum_{\omega\in\Omega}\exp(-2\beta N^{ih}(\omega))} = 0 & (4.52)
\end{aligned}
$$

with $N^{ih}(\widehat{\omega}) = \sum_{\{s,r\}\in\mathcal{C}} \delta(\widehat{\omega}_s, \widehat{\omega}_r)$ is the number of inhomogeneous cliques in $\widehat{\omega}$. From Equation (4.52), we get

$$
N^{ih}(\widehat{\omega}) = \frac{\sum_{\omega\in\Omega} N^{ih}(\omega)\exp(-2\beta N^{ih}(\omega))}{\sum_{\omega\in\Omega}\exp(-2\beta N^{ih}(\omega))} \tag{4.53}
$$

The right hand side is also called the *energy mean*. Since $\ln(Z(\beta))$ is *convex* in Θ [9, 45], the gradient can be approximated by stochastic relaxation [45]:

Algorithm 4.5.1 (Hyperparameter Estimation)

① *Set $k = 0$, initialize $\widehat{\beta}^0$ and let $N^{ih}(\widehat{\omega})$ denote the number of inhomogeneous cliques in the estimate of the labeling.*

② *Using SA at a fixed temperature T, generate a new labeling η, sampling from*

$$P(\mathcal{X} = \omega) = \frac{\exp\left(-\frac{\widehat{\beta}^k}{T}\sum_{\{s,r\}\in\mathcal{S}}\delta(\omega_s, \omega_r)\right)}{Z(\widehat{\beta}^k)}. \tag{4.54}$$

Compute the number of inhomogeneous cliques $N^{ih}(\eta)$ in η.

③ *If $N^{ih}(\eta) \approx N^{ih}(\widehat{\omega})$ then stop, else $k = k + 1$. If $N^{ih}(\eta) < N^{ih}(\widehat{\omega})$ then decrease $\widehat{\beta}^k$, if $N^{ih}(\eta) > N^{ih}(\widehat{\omega})$ then increase $\widehat{\beta}^k$, and goto Step ②.*

The complete parameter estimation process is the following: Given an image $\mathcal{F}$, compute the histogram and initialize the mean and the deviation of the classes, using Algorithm 4.4.1 for instance. Then, using one of the iterative algorithms described in Section 4.3, estimate Θ. Once the final estimate $\widehat{\Theta}$ of the parameters is obtained, one proceeds to the ordinary segmentation with known parameters. A possible formulation of an unsupervised segmentation algorithm (the one used for the simulations) is the following:

Algorithm 4.5.2 (Unsupervised Segmentation)

① *Given an image $\mathcal{F}$, compute its histogram and for each $\lambda \in \Lambda$, initialize μ_λ and σ_λ using Algorithm 4.4.1. β is initialized in an ad-hoc way.*

② **(Estimation)** *Using Algorithm 4.3.4 (ICE), get an estimate $\widehat{\Theta}$ of the parameters.*

③ **(Segmentation)** *Given the parameters $\widehat{\Theta}$, do an ordinary supervised segmentation to get the MAP estimate of the label field given $\mathcal{F}$ and $\widehat{\Theta}$.*

4.5.1 Parameter Estimation of a Hierarchical MRF Model

Considering the segmentation model presented in Section 2.6.4, we have the following logarithmic likelihood function:

$$\sum_{i=0}^{M} \sum_{s^i \in \mathcal{S}^i} \sum_{s \in b^i_{s^i}} \left(-\ln(\sqrt{2\pi}\sigma_{\omega_s}) - \frac{(f_s - \mu_{\omega_s})^2}{2\sigma^2_{\omega_s}} \right)$$

$$-2\beta \underbrace{\sum_{i=0}^{M} q^i \sum_{C^i \in \mathcal{C}^i} \delta(\widehat{\omega}_{C^i})}_{N^{ih}(\widehat{\omega})} - 2\gamma \underbrace{\sum_{C \in \bar{\mathcal{C}}_3} \delta(\widehat{\omega}_C)}_{\bar{N}^{ih}(\widehat{\omega})} - \ln(Z(\beta, \gamma)) \tag{4.55}$$

where q^i is the number of cliques between two neighboring blocks at scale $\mathcal{B}^i$ (for more details, see Section 2.5.3). $N^{ih}(\widehat{\omega})$ denotes the number of inhomogeneous cliques siting at the same scale and $\bar{N}^{ih}(\widehat{\omega})$ denotes the number of inhomogeneous cliques siting astride two neighboring levels in the pyramid. First, let us consider the first term:

$$\sum_{i=0}^{M} \sum_{s^i \in \mathcal{S}^i} \sum_{s \in b^i_{s^i}} \left(-\ln(\sqrt{2\pi}\sigma_{\omega_s}) - \frac{(f_s - \mu_{\omega_s})^2}{2\sigma^2_{\omega_s}} \right)$$

$$= \sum_{\lambda \in \Lambda} \sum_{i=0}^{M} \sum_{s^i \in \mathcal{S}^i_\lambda} \sum_{s \in b^i_{s^i}} \left(-\ln(\sqrt{2\pi}\sigma_\lambda) - \frac{(f_s - \mu_\lambda)^2}{2\sigma^2_\lambda} \right) \tag{4.56}$$

where $\mathcal{S}^i_\lambda$ is the set of sites at level i where $\widehat{\omega}_{s^i} = \lambda$. Derivating with respect to μ_λ and σ_λ, we get:

$$\begin{aligned} \forall \lambda \in \Lambda: \quad \mu_\lambda &= \frac{1}{\sum_{i=0}^{M} \mid \mathcal{S}^i_\lambda \mid} \sum_{i=0}^{M} \sum_{s^i \in \mathcal{S}^i_\lambda} \sum_{s \in b^i_{s^i}} f_s \\ \sigma^2_\lambda &= \frac{1}{\sum_{i=0}^{M} \mid \mathcal{S}^i_\lambda \mid} \sum_{i=0}^{M} \sum_{s^i \in \mathcal{S}^i_\lambda} \sum_{s \in b^i_{s^i}} (f_s - \mu_\lambda)^2 \end{aligned} \tag{4.57}$$

Notice that a grey-level value f_s may be considered several times. More precisely, f_s is considered m-times in the above sum for a given λ if there is m scale where $\widehat{\omega}$ assigns the label λ to the site s. m can also be seen as a weight. Obviously, the more s has been labeled by λ at different levels, the more is probable that s belongs to class λ and hence its grey-level value f_s characterizes better the class λ. The derivate of the logarithmic likelihood function with respect to β and γ is given by

$$\frac{\partial}{\partial\beta}\left(-2\beta N^{ih}(\widehat{\omega}) - \ln(Z(\beta,\gamma))\right) = -N^{ih}(\widehat{\omega}) - \frac{\partial}{\partial\beta}\ln(Z(\beta,\gamma)) \tag{4.58}$$

$$\frac{\partial}{\partial\gamma}\left(-2\gamma \bar{N}^{ih}(\widehat{\omega}) - \ln(Z(\beta,\gamma))\right) = -\bar{N}^{ih}(\widehat{\omega}) - \frac{\partial}{\partial\gamma}\ln(Z(\beta,\gamma)) \tag{4.59}$$

From which, we get

$$N^{ih}(\widehat{\omega}) = \frac{\sum_{\omega\in\Omega} N^{ih}(\omega)\exp(-2\beta N^{ih}(\omega) - 2\gamma \bar{N}^{ih}(\omega))}{\sum_{\omega\in\Omega}\exp(-2\beta N^{ih}(\omega) - 2\gamma \bar{N}^{ih}(\omega))} \tag{4.60}$$

$$\bar{N}^{ih}(\widehat{\omega}) = \frac{\sum_{\omega\in\Omega} \bar{N}^{ih}(\omega)\exp(-2\beta N^{ih}(\omega) - 2\gamma \bar{N}^{ih}(\omega))}{\sum_{\omega\in\Omega}\exp(-2\beta N^{ih}(\omega) - 2\gamma \bar{N}^{ih}(\omega))} \tag{4.61}$$

The solution of the above equations, as in the monogrid case, can be obtained using Algorithm 4.5.1 with some modifications:

Algorithm 4.5.3 (Hierarchical Hyperparameter Estimation)

① *Set $k = 0$ and initialize $\widehat{\beta}^0$ and $\widehat{\gamma}^0$. Furthermore, let $N^{ih}(\widehat{\omega})$ denote the number of inhomogeneous cliques at the same scale and $\bar{N}^{ih}(\widehat{\omega})$ denotes the number of inhomogeneous cliques between levels.*

② *Using SA at a fixed temperature T, generate a new labeling η sampling from*

$$P(\mathcal{X}=\omega) = \frac{\exp\left(-\frac{\widehat{\beta}^k}{T}\sum_{i=0}^{M}\sum_{\{s,r\}\in\mathcal{C}^i}\delta(\omega_s,\omega_r) + \frac{\widehat{\gamma}^k}{T}\sum_{\{s,r\}\in\bar{\mathcal{C}}}\delta(\omega_s,\omega_r)\right)}{Z(\widehat{\beta}^k,\widehat{\gamma}^k)}. \tag{4.62}$$

Compute the number of inhomogeneous cliques $N^{ih}(\eta)$ and $\bar{N}^{ih}(\eta)$ in η.

③ *If $N^{ih}(\eta) \approx N^{ih}(\widehat{\omega})$* and *$\bar{N}^{ih}(\eta) \approx \bar{N}^{ih}(\widehat{\omega})$ then stop, else $k = k+1$. If $N^{ih}(\eta) < N^{ih}(\widehat{\omega})$ then decrease $\widehat{\beta}^k$, if $N^{ih}(\eta) > N^{ih}(\widehat{\omega})$ then increase $\widehat{\beta}^k$. $\widehat{\gamma}^k$ is obtained in the same way. Continue Step ② with $(\widehat{\beta}^k, \widehat{\gamma}^k)$.*

Algorithm 4.5.2 can also be applied to the hierarchical model with trivial modifications. Here we give the algorithm used for the simulations:

Algorithm 4.5.4 (Unsupervised Hierarchical Segmentation)

① *Given an image $\mathcal{F}$, compute its histogram and for each $\lambda \in \Lambda$, initialize μ_λ and σ_λ using Algorithm 4.4.1. β and γ is initialized in an ad-hoc way.*

② **(Estimation)** *Using Algorithm 4.3.3 (ASA), get an estimate $\widehat{\Theta}$ of the parameters.*

③ **(Segmentation)** *Given the parameters $\widehat{\Theta}$, do an ordinary supervised segmentation to get the MAP estimate of the label field given $\mathcal{F}$ and $\widehat{\Theta}$.*

We remark, that in Step ②, the Gaussian parameters were computed considering only the finest level (cf. Equation (4.51)) and not the entire pyramid (cf. Equation (4.57)).

4.6 Experimental Results

We have tested the proposed monogrid and hierarchical unsupervised algorithms on noisy synthetic and real images. The algorithms were implemented on a Connection Machine CM200 [65] (for more details, see Section 2.7.1). We have compared the obtained parameters and segmentation results to the supervised results already presented in Chapter 2. In general, the quality of unsupervised results are as good, or sometimes slightly better, than the results of supervised segmentation. We observed, however, that the unsupervised algorithm is more sensitive to noise than the supervised one. This is due to the initialization, in particular the initialization of the mean and the variance of the classes (the initialization of β and γ are not crucial). For example, in the case of the "triangle" image with SNR= $3dB$ one class has been lost, but with SNR= $5dB$ the result is as good as for the supervised algorithm.

Before evaluating the results, let us explain some important points of the implementation. The only parameter which has to be defined by the user is the number of classes (or regions). All the other parameters are estimated automatically from the data. Essentially, we have followed Algorithm 4.5.2: First, the initial values of the mean and variance have been estimated using Algorithm 4.4.1. For the hyperparameters, we have chosen as initial values $\beta = 0.7$ and $\gamma = 0.1$. Experiments show that these initial values are not vital, practically any value between 0.5 and 1 is good for β and a value close to zero is good for γ.

In the next step (Step ② of Algorithm 4.5.2), we use the ICE algorithm (see Algorithm 4.3.4) to iteratively reestimate the parameters. We have chosen ICM to generate labelings because of its rapidity: Given the parameters $\widehat{\Theta}^n$, the ICM is used to maximize the a posteriori probability of the label field ω. Suppose, that ICM converges in N iterations (N is typically less than 10) given N realizations of ω. Using these labelings, we have to compute N ML estimates of Θ (see Algorithm 4.3.4 for more details).

For the hierarchical model, however, we have used ASA (cf. Algorithm 4.5.4), because the ICE algorithm would be too slow with the hierarchical model: Using ICM, we maximize the a posteriori probability of ω, given the parameter estimates $\widehat{\Theta}^n$. Then the ML estimate is computed based on the obtained labeling. Another modification is that the Gaussian parameters were computed considering *only the finest level* and not the entire pyramid as explained in Section 4.5.1. This is because the variances obtained with the original algorithm were to large. This modification also reduces the computing time.

Once the sequence $\widehat{\Theta}^n$ becomes steady, the estimation step is finished and one proceeds to the segmentation (with known parameters) using the Gibbs sampler, for instance.

The algorithms were tested on the "checkerboard" (Figure 4.2), "triangle" (Figure 4.3) and "holland" (Figure 4.4–Figure 4.6) images. For the synthetic images, we also give the histogram, since the initial estimates are based on it (the histogram of the "holland" image can be found in Figure 2.18). In Table 4.2, Table 4.4 and Table 4.6, we compare the parameters obtained by the unsupervised algorithm to the ones used for the supervised segmentation. We remark that the parameters of the supervised algorithm are not necessarily correct. They have been computed on training sets selected by an expert (cf. Figure 4.4) using the algorithm described in Section 2.3. In Table 4.3, Table 4.5 and Table 4.7, we give the computer time of the estimation and segmentation. As we can see, the estimation requires much more time than the segmentation. The hyperparameter estimation requires the biggest part of the computer time since it consists of generating new labelings by SA in Step ② of Algorithm 4.5.1.

Table 4.1 provides an objective comparison of supervised and unsupervised segmentation results based on the number of misclassified pixels. The obtained results are practically the same for supervised and unsupervised segmentation.

In summary, the presented algorithms provide results comparable to supervised segmentations, but they require much more computing time, and they are slightly more sensitive to noise. The main advantage is, of course, that unsupervised methods are completely data-driven. The only input parameter is the number of regions. We believe that, for unsupervised methods, the main problem is still the initialization of the Gaussian parameters. Hence, a natural extension of the work presented in this chapter would be to look for more efficient initialization techniques.

Appendix

4.A Images

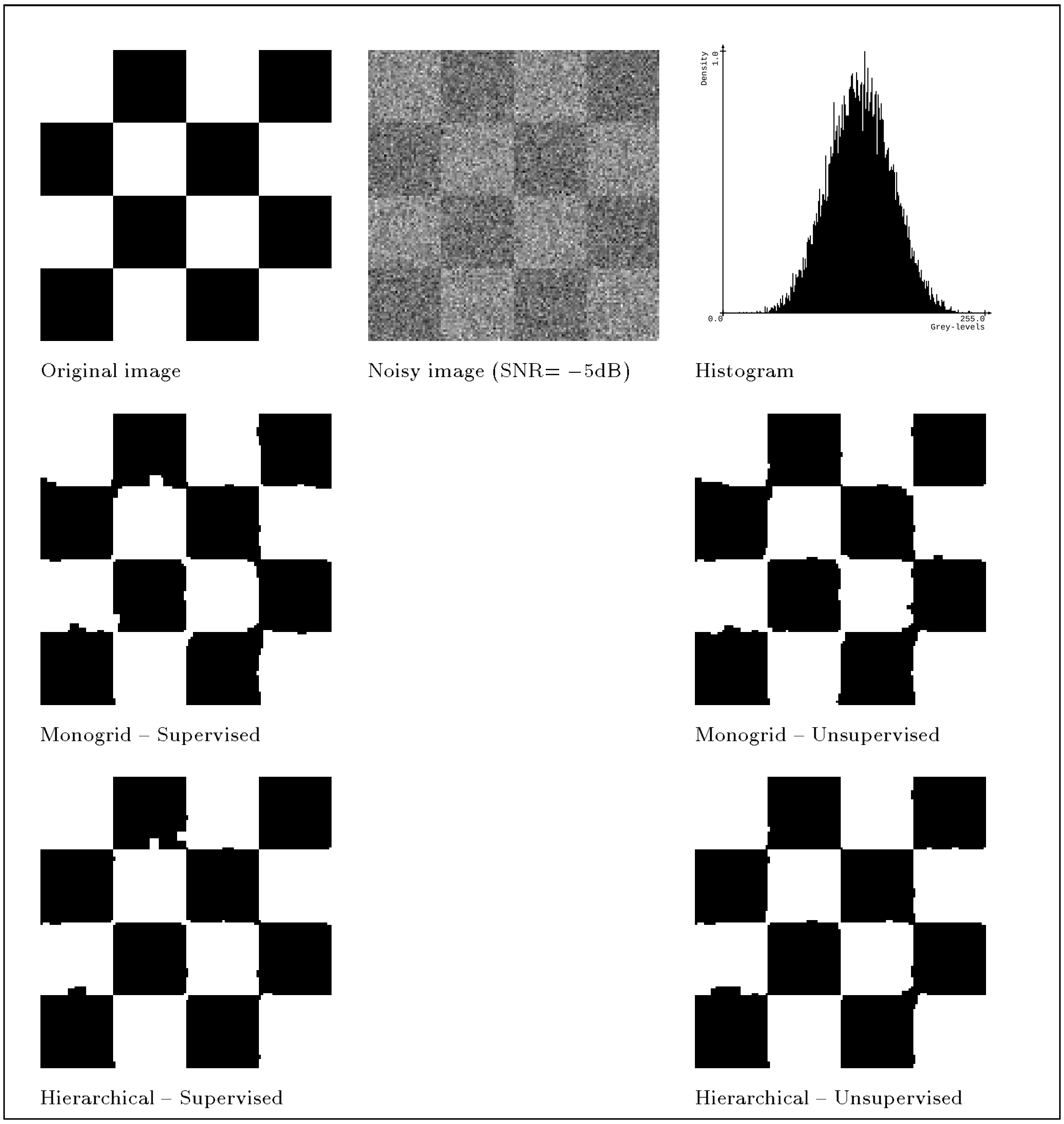

Figure 4.2: *Supervised and unsupervised segmentation results on the "checkerboard" image with 2 classes (Gibbs Sampler).*

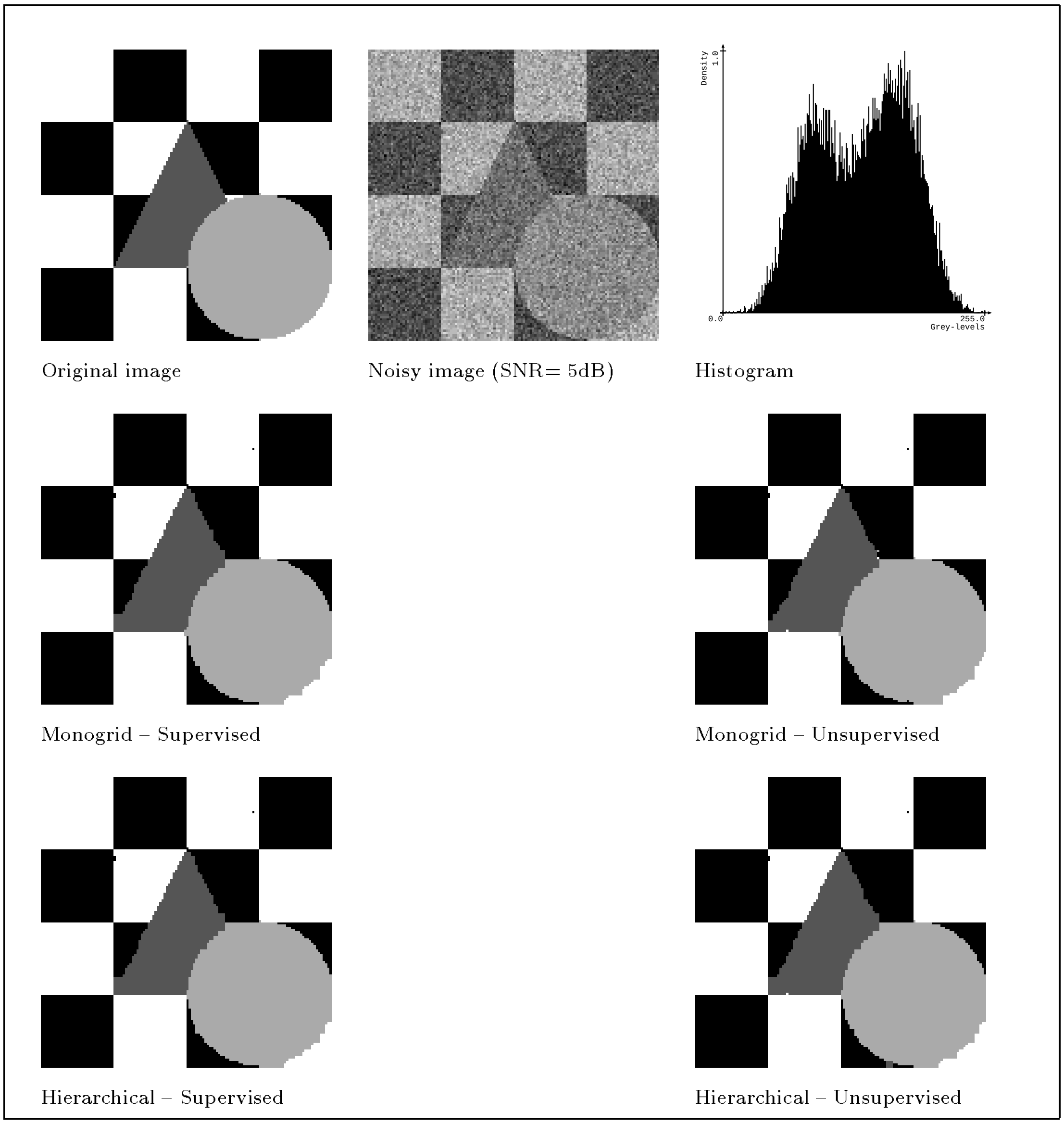

Figure 4.3: *Supervised and unsupervised segmentation results on the "triangle" image with 4 classes (Gibbs Sampler).*

Figure 4.4: *Training areas on the "holland" image.*

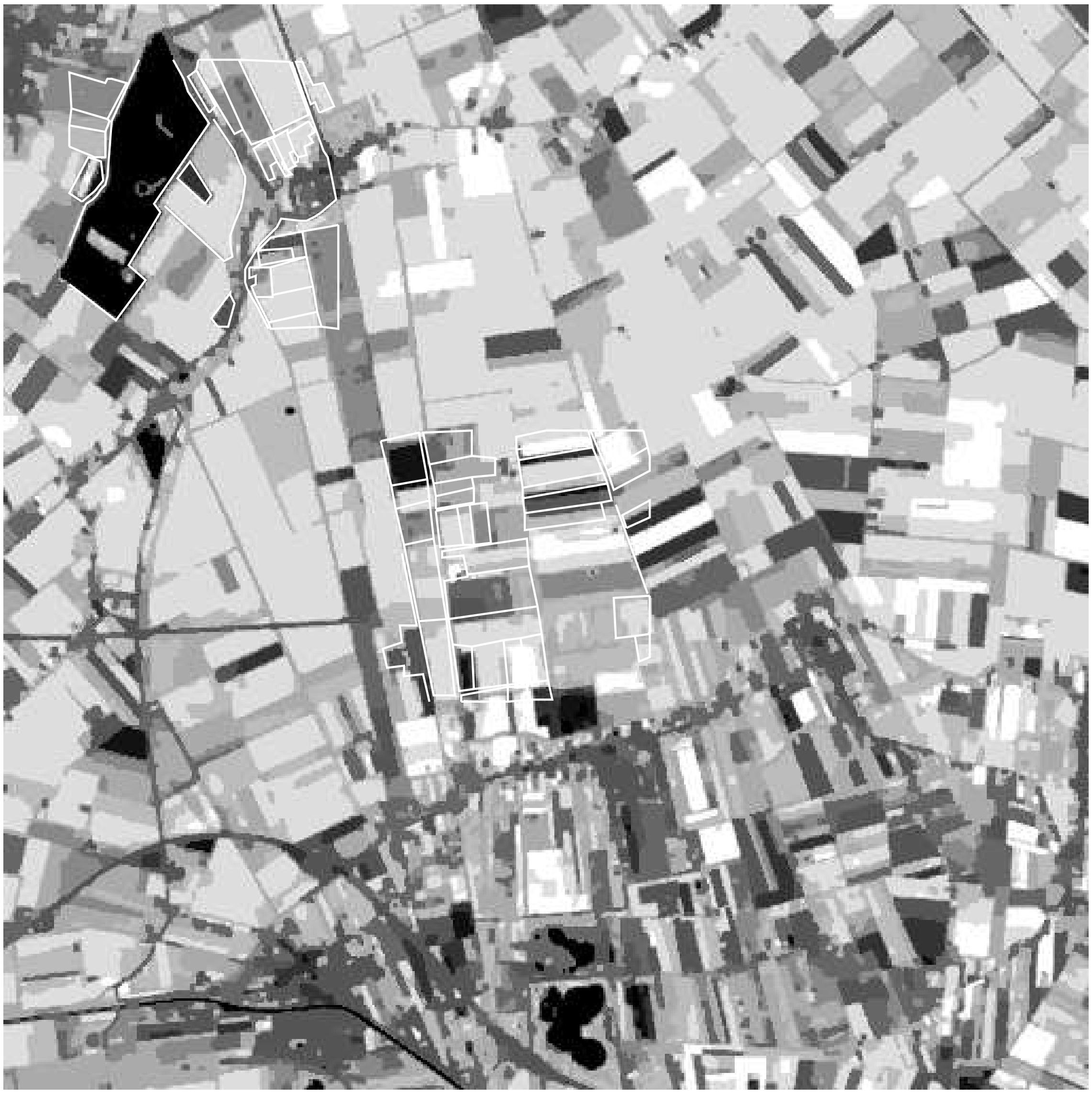

Figure 4.5: *Supervised segmentation result with* 10 *classes (Gibbs Sampler).*

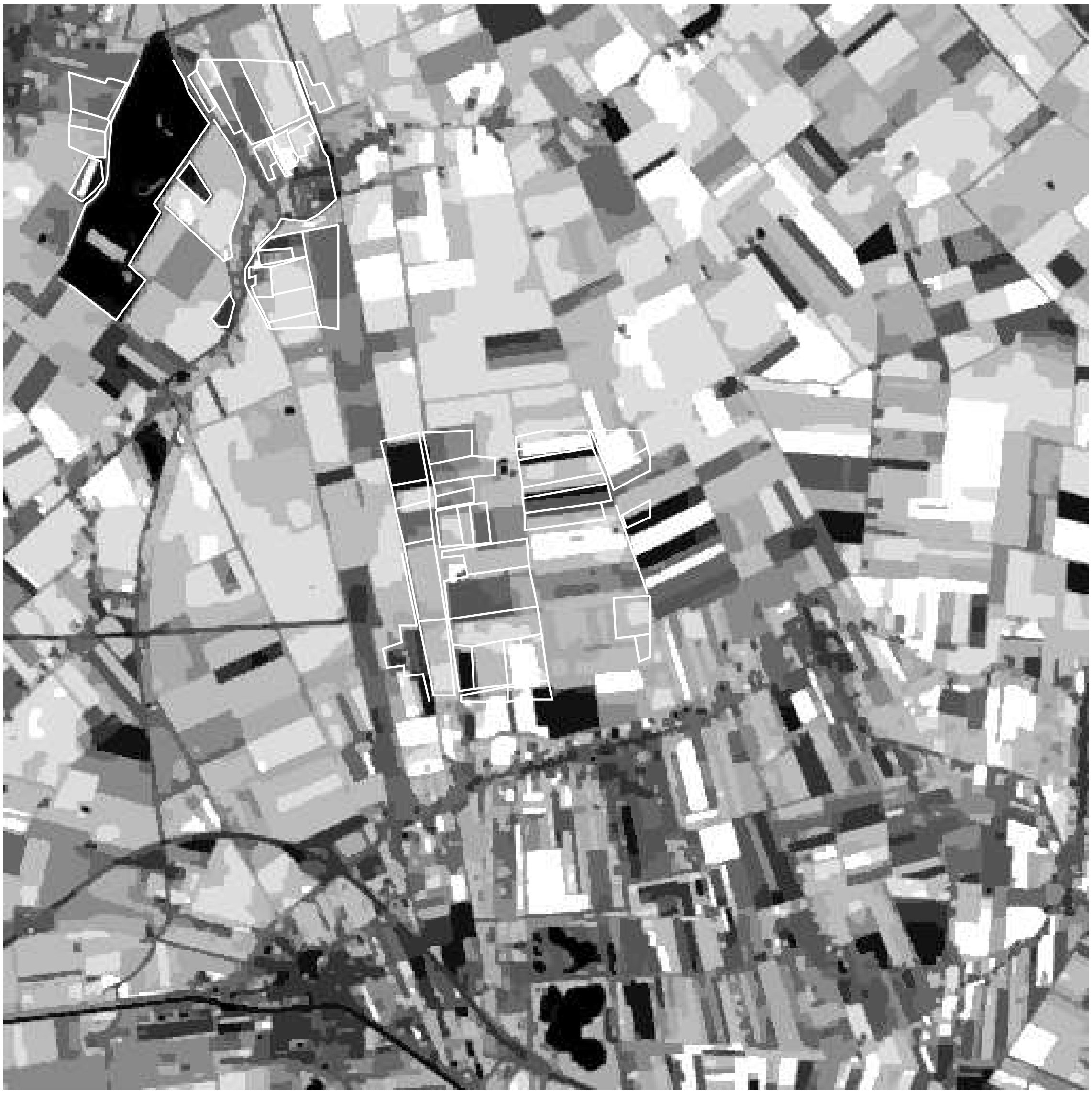

Figure 4.6: *Unsupervised segmentation result with* 10 *classes (Gibbs Sampler).*

4.B Tables

Model	Image	Supervised	Unsupervised
Monogrid	checkerboard	260 (1.59%)	213 (1.41%)
	triangle	112 (0.68%)	103 (0.63%)
Hierarchical	checkerboard	115 (0.7%)	147 (0.9%)
	triangle	104 (0.63%)	111 (0.68%)

Table 4.1: *Comparison of the number of misclasified pixels.*

Monogrid model			
	Unsupervised		
Parameter	Initial	Final	Supervised
μ_0	123.5	117.3	119.2
σ_0^2	256.0	680.0	659.5
μ_1	170.0	151.5	149.4
σ_1^2	169.0	668.2	691.4
β	0.7	0.7	0.9

Hierarchical model			
	Unsupervised		
Parameter	Initial	Final	Supervised
μ_0	123.5	126.7	119.2
σ_0^2	256.0	903.4	659.5
μ_1	170.0	151.5	149.4
σ_1^2	169.0	689.3	691.4
β	0.7	0.7	0.7
γ	0.1	0.1	0.3

Table 4.2: *Parameters of the "checkerboard" image.*

Model	VPR	Total CPU time	Estimation	Segmentation
Monogrid	2	142.73 sec.	133.57 sec.	9.16 sec.
Hierarchical	4	1551.93 sec.	1042.46 sec.	446.52 sec.

Table 4.3: *Computer times of the "checkerboard" image.*

Monogrid model			
	Unsupervised		
Parameter	Initial	Final	Supervised
μ_0	83.5	84.3	85.48
σ_0^2	256.0	480.5	446.60
μ_1	100.0	117.3	115.60
σ_1^2	169.0	416.3	533.97
μ_2	152.5	148.1	146.11
σ_2^2	676.0	457.8	540.32
μ_3	181.5	178.5	178.01
σ_3^2	100.0	490.9	504.34
β	0.7	1.0	1.0

Hierarchical model			
	Unsupervised		
Parameter	Initial	Final	Supervised
μ_0	83.5	84.3	85.48
σ_0^2	256.0	483.9	446.60
μ_1	100.0	115.5	115.60
σ_1^2	169.0	444.6	533.97
μ_2	152.5	146.7	146.11
σ_2^2	676.0	502.1	540.32
μ_3	181.5	177.9	178.01
σ_3^2	100.0	500.0	504.34
β	0.7	1.0	0.7
γ	0.1	0.1	0.1

Table 4.4: *Parameters of the "triangle" image.*

Model	VPR	Total CPU time	Estimation	Segmentation
Monogrid	2	249.75 sec.	237.00 sec.	12.75 sec.
Hierarchical	4	1762.23 sec.	1232.82 sec.	529.41 sec.

Table 4.5: *Computer times of the "triangle" image.*

Parameter	Unsupervised Initial	Unsupervised Final	Supervised
μ_0	51.5	53.1	54.6
σ_0^2	36.0	10.3	93.1
μ_1	60.0	77.2	73.5
σ_1^2	49.0	64.3	4.1
μ_2	70.5	89.6	82.5
σ_2^2	49.0	30.7	35.5
μ_3	80.5	102.5	93.8
σ_3^2	64.0	35.7	93.7
μ_4	97.5	116.2	100.5
σ_4^2	441.0	27.6	308.8
μ_5	122.5	127.2	122.8
σ_5^2	484.0	18.9	8.9
μ_6	136.0	138.6	129.9
σ_6^2	1.0	20.2	37.4
μ_7	152.5	152.7	146.6
σ_7^2	625.0	18.0	15.3
μ_8	169.0	162.4	159.9
σ_8^2	1.0	7.4	31.3
μ_9	181.5	174.2	182.3
σ_9^2	25.0	54.1	73.1
β	0.7	1.3	1.0

Table 4.6: *Parameters of the "holland" image.*

Model	VPR	Total CPU time	Estimation	Segmentation
Monogrid	32	3576.58 sec.	3270.78 sec.	305.81 sec.

Table 4.7: *Computer times of the "holland" image.*

Conclusion

We have reviewed the three main steps of statistical image processing in early vision: modelization, optimization and parameter estimation. We have considered low level vision tasks in a common framework, called image labeling, where the problem is reduced to assigning labels to pixels. Our approach is probabilistic, using Markov Random Fields (MRF) and Bayesian estimation, in particular Maximum A Posteriori (MAP) estimation. The advantage of MRF modelization is that a priori information can be "coded" *locally* through clique potentials. We have also discussed pyramidal MRF models, which reduce the computing time and increase the quality of final results. Parameter estimation is an important problem in real-life applications in order to implement completely data-driven algorithms. We have adapted some existing methods to the estimation of monogrid and hierarchical model-parameters. The preliminary results are encouraging but there is still a lot of work to do.

All the MRF models we have considered result in a non-convex energy function. The minimization of this function is done by Simulated Annealing or deterministic relaxation. We have also discussed the possible parallelization techniques of optimization algorithms.

Our main result is a new hierarchical MRF model and a Multi-Temperature Annealing algorithm proposed for the energy minimization of the model. The convergence of the MTA algorithm has been proved towards a *global* optimum in the most general case, where each clique may have its own *local* temperature schedule. There is still some open problems such as the relationship between monogrid and hierarchical MAP estimates, implementing the hierarchical model on a pyramidal computer, or looking for a faster parameter estimation method.

Summary

We have reviewed in this thesis the three main steps of statistical image processing in early vision:

- Modelization
- Optimization
- Parameter estimation

For the *modelization*, we have considered a lot of low level vision tasks which can be formulated in a common framework, called image labeling, where the problem is reduced to assigning labels to pixels. We have proposed to solve this problem in a probabilistic framework using Markov Random Fields (MRF) and Bayesian estimation, in particular Maximum A Posteriori (MAP) estimation. The advantage of MRF modelization is that it requires little a priori information about the "world" model, and these informations can be defined *locally* through clique potentials. Another advantage is that the local behavior of MRF's permits to develop highly parallel algorithms. Unfortunately, even with massively parallel algorithms, finding the MAP estimate is very time consuming due to the non-convexity of the energy function. To eliminate this drawback, many authors propose multigrid pyramidal MRF schemes. The advantage of such an approach is that at coarser resolution, the configuration space is considerably smaller and thus the optimization problem becomes easier. Using a top-down relaxation strategy in the pyramid, computing time can be considerably reduced and the quality of final results are increased with respect to monogrid schemes.

In the *optimization* step, we have to minimize a non-convex energy function in order to find the MAP estimate. Basically, we have two choices: *stochastic* or *deterministic* relaxation. The former is guaranteed to find *global* minima but it requires a large amount of computing time. Deterministic methods aim at finding a reasonably good approximation of the optimum. The obtained result is always a *local* minimum, but they are less time-consuming than stochastic algorithms. Parallelization is another possible way to speed up optimization algorithms. The convergence of some parallelization schemes has been proved (especially for such algorithms where the conditions of the convergence of sequential algorithms have not been violated.) but there are some interesting scheme for which the convergence has not yet been proved: Graffigne's parallelized Markov chains method or massive parallelization at rate τ proposed by Azencott (in the latter case, convergence is proved but it is not shown that the limit is a global optimum).

In real life applications, the parameters are usually unknown, they have to be estimated from the data. In such cases, the first step is the *parameter estimation*. Developing a completely data-driven algorithm is an extremely difficult problem. We have presented some iterative algorithms and implemented an unsupervised segmentation algorithm. The first results, presented in Chapter 4 are encouraging but we have observed that unsupervised algorithms require much more computing time due to the hyperparameter estimation (β and γ). In the current implementation, they are computed using Simulated Annealing, which is very time-consuming. Another important point is the initialization of the Gaussian parameters for each class. We have noticed that unsupervised algorithms are more sensitive to noise than supervised ones. This sensitivity is due to the bad initialization in the case of noisy images.

Results and Future Research Directions

Our main result is a new hierarchical MRF model and a Multi-Temperature Annealing algorithm proposed to the energy minimization. The hierarchical model is based on a multiscale one, proposed by Perez *et al.* : we use the same procedure to derive coarser grids but we have introduced a new inter-level communication scheme. The so obtained model is a fully connected MRF over the whole pyramid. It has some advantages but also some drawbacks: the new connections allow to propagate local interactions more efficiently, giving better estimates (in particular for fast deterministic relaxation algorithms, such as ICM). On the other hand, these interactions make the model more complex, demanding much more computing time. We note, however, that the model was implemented on a Connection Machine, which is not an optimal architecture for pyramidal models (see Section 2.7.1 for more details). We believe that on a pyramid architecture, better computing times could be achieved. A future work would be to implement the model on a pyramidal computer.

Another interesting subject would be to study the relationship between the monogrid MAP estimate and the hierarchical MAP estimate. An often criticized point of the proposed model is that the energy function is defined on the whole pyramid but, at the convergence, we only take the finest level as the final result and there is no guarantee that it is also optimal. However, defining an energy function over a MRF and then taking only a part of it as the final result is not a new idea. We can mention the line process introduced by Geman and Geman in their image restoration model as an example[2]. Intuitively, it is clear that the hierarchical model incorporates cliques

[2]They have defined an energy function both on the line process and the pixel process. Then the energy function has been optimized but only the pixel process has been taken as the result.

with far apart sites. Considering a pyramid where the coarsest layer contains only one pixel, the projection of this pixel to the finest level, keeping all the interactions, would result in a completely connected monogrid MRF model. It would be interesting to study this hypothesis in a rigorous mathematical framework.

The Multi-Temperature Annealing has originally been proposed to solve the optimization problem of hierarchical models, but it is a more general scheme. In fact, the mathematical study of the algorithm is quite general and does not suppose a pyramidal structure. The convergence of the algorithm has been proved towards a *global* optimum in the most general case, where each clique may have its own *local* temperature schedule. It would be interesting to apply MTA to other, non hierarchical models.

Parameter estimation still requires a lot of work. The results presented in Chapter 4 are only first preliminary results, especially for the hierarchical model. We see two main problems which have to be studied in details. The first one is to find a better algorithm to estimate the hyperparameters. The presented algorithm uses Simulated Annealing which is very time consuming. May be, Mean Field approximation would result in a faster convergence [86, 115]. Another problem is the initialization of the Gaussian parameters: The actual implementation is more sensitive to noise than supervised algorithms. We need a method which is less dependent on the initialization or a better initialization algorithm to get rid of this weakness in the current algorithm.

Publications

1. Z. Kato, M. Berthod, J. Zerubia, W. Pieczynski: Unsupervised Adaptive Image Segmentation. *ICASSP'95, Detroit, Mitchigan, USA, May 1995.*

2. Z. Kato, M. Berthod, J. Zerubia: A hierarchical Markov random Field Model and Multi-Temperature Annealing for parallel image classification. *Submitted to CVGIP.*

3. J. Zerubia, Z. Kato, M. Berthod: Multi-Temperature Annealing: a new approach for the energy-minimization of hierarchical Markov Random Field models. *ICPR-Computer Vision and Applications, Jerusalem, Oct. 1994.*

4. Z. Kato, M. Berthod, J. Zerubia: A hierarchical Markov random Field Model and Multi-Temperature Annealing for parallel image classification. *Research Report 1938, INRIA, Aug. 1993.*

5. Z. Kato, M. Berthod, J. Zerubia: A Hierarchical Markov Random Field Model for Image Classification. *In Proc.* 8^{th} *IMDSP Workshop, Cannes, France, September 1993.*

6. Z. Kato, M. Berthod, J. Zerubia: Multi-scale Markov Random Field models for parallel image classification. *In Proc. ICCV, Berlin, May 1993.*

7. Z. Kato, M. Berthod, J. Zerubia: Parallel image classification using multi-scale Markov Random Fields. *In Proc. ICASSP, Minneapolis, Apr. 1993.*

8. M. Berthod, Z. Kato, J. Zerubia: DPA: A Deterministic Approach to the MAP. *Accepted for publication in IEEE Trans. on Image Processing*, 1994.

9. Z. Kato, J. Zerubia, M. Berthod: Bayesian image classification using Markov Random Fields. In A. M.-D. G. Demoments, editor, *Maximum Entropy and Bayesian Methods*, pages 375–382. Kluwer Academic Publisher, 1993.

10. Z. Kato, J. Zerubia, M. Berthod: Image classification using Markov Random Fields with two new relaxation methods: Deterministic Pseudo Annealing and Modified Metropolis Dynamics. *Research Report 1606, INRIA, Feb. 1992.*

11. Z. Kato, J. Zerubia, M. Berthod: Satellite image classification using a Modified Metropolis Dynamics. *In Proc. ICASSP, San-Francisco, California, USA, Mar. 1992.*

Bibliography

[1] O. Allagnat, J. M. Boucher, D. C. He, and W. Pieczynski. Hidden Markov Fields and Unsupervised Segmentation of Images. In *Proc. ICPR'92*, 1992.

[2] R. Azencott. Image Analysis and Markov Fields. In *Proc. ICIAM'87*, pages 53–61, Paris, France, June 29-July 3 1987.

[3] R. Azencott. Markov fields and image analysis. *Proc. AFCET, Antibes*, 1987.

[4] R. Azencott. Parallel Simulated Annealing: An Overview of Basic Techniques. In R. Azencott, editor, *Simulated Annealing: Parallelization Techniques*, pages 37–46. John Wiley & Sons, Inc., 1992.

[5] R. Azencott, editor. *Parallel Simulated Annealing. Parallelization Techniques.* John Wiley & Sons, Inc., 1992.

[6] R. Azencott and C. Graffigne. Parallel Annealing by Periodically Interacting Multiple Searches: Acceleration Rates. In R. Azencott, editor, *Simulated Annealing: Parallelization Techniques*, pages 81–90. John Wiley & Sons, Inc., 1992.

[7] N. Balram and J. F. Moura. Parameter Estimation in 2D Fields. In *Proc. ICASSP'92*, pages III–345–348, San Francisco, USA, March 1992.

[8] Y. Bard. *Nonlinear Parameter Estimation.* Academic Press, Inc., 1974.

[9] R. J. Baxter. *Exactly Solved Models in Statistical Mechanics.* Academic Press, 1990.

[10] M. Berthod, G. Giraudon, and J. Stromboni. Determistic Pseudo-Annealing : a Suboptimal Scheme for optimization in Markov Random Fields. An application to pixel classification. In *Proc. ECCV*, Santa Margherita Ligure, Italy, May 1992.

[11] M. Berthod, Z. Kato, and J. Zerubia. DPA: A Deterministic Approach to the MAP. *Accepted for publication in IEEE Trans. on Image Processing*, 1995.

[12] J. Besag. Efficiency of pseudolikelihood estimation for simple Gaussian fields. *Biometrika*, 64(3):616–618, 1977.

[13] J. Besag. On the statistical analysis of dirty pictures. *Jl. Roy. Statis. Soc. B.*, 1986.

[14] J. E. Besag. Spatial interaction and the statistical analysis of lattice systems (with discussion). *Jl. Roy. Statis. Soc. B. 36.*, pages 192–236, 1974.

[15] A. Blake and A. Zisserman. Visual reconstruction. *MIT Press, Cambridge - MA*, 1987.

[16] C. Bouman. A Multiscale Image Model for Bayesian Image Segmentation. Technical Report TR-EE 91-53, Purdue University, 1991.

[17] C. Bouman and B. Liu. Multiple Resolution segmentation of Texture Images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13:99–113, 1991.

[18] C. Bouman and M. Shapiro. Multispectral Image Segmentation using a Multiscale Model. In *Proc. ICASSP'92*, San Francisco, USA, March 1992.

[19] B. Braathen, W. Pieczynski, and P. Masson. Global and Local Methods of Unsupervised Bayesian Segmentation of Images. *Machine Graphics and Vision*, 2(1):39–52, 1993.

[20] G. L. Bretthorst. An Introduction to Parameter Estimation Using Bayesian Probability Theory. In P. F. Fougère, editor, *Maximum Entropy and Bayesian Methods*, Netherlands, 1990. Kluwer Academic Publishers.

[21] H. Caillol, A. Hillion, and W. Pieczynski. Fuzzy Random Fields and Unsupervised Image Segmentation. *IEEE Geoscience and Remote Sensing*, 31(4):801–810, July 1993.

[22] O. Catoni and A. Trouvé. Parallel Annealing by Multiple Trials. In R. Azencott, editor, *Simulated Annealing: Parallelization Techniques*, pages 129–144. John Wiley & Sons, Inc., 1992.

[23] G. Celeux. L'algorithme sem: un algorithme d'apprentissage probabiliste pour la reconnaissance de mélange de densité. *Revue de statistique appliqué*, 34(2):35–52, 1986.

[24] V. Černy. Thermodynamical Approach to the Traveling Salesman Problem: An Efficient Simulation Algorithm. *J. Opt. Theory Appl.*, 45(1):41–51, January 1985.

[25] B. Chalmond. Image restoration using an estimated markov model. *Signal Processing*, 15:115–129, 1988.

[26] B. Chalmond. An Iterative Gibbsian Technique for Reconstruction of M-ary Images. *Pattern Recognition*, 22(6), 1989.

[27] P. Chou and C. Brown. The theory and practice of bayesian image labeling. *International Journal of Computer Vision*, 4:185–210, 1990.

[28] F. S. Cohen and D. B. Cooper. Simple Parallel Hierarchical and Relaxation Algorithms for Segmenting Noncausal Markov Random Fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(2):195–219, March 1987.

[29] G. R. Cross and A. K. Jain. Markov Random Field Texture Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(1):25–39, January 1983.

[30] I. Csiszár and J. Körner. *Information Theory. Coding Theorems for Discrete Memoryless Systems.* Probability and Mathematical Statistics. Academic Press Inc., 1981.

[31] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum Likelihood from Incomplete Data via the EM Algorithm. *J. Roy. Statist. Soc., ser. B*, vol. 39(1):1–38, 1977.

[32] H. Derin. Estimation Components of Univariate Gaussian Mixtures Using Prony's Method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(1):142–148, January 1987.

[33] H. Derin and H. Elliott. Modeling and Segmentation of Noisy and Textured Images Using Gibbs Random Fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(1):39–55, January 1987.

[34] H. Derin, H. Elliott, R. Cristi, and D. Geman. Bayes Smoothing Algorithms for Segmentation of Binary Images Modeled by Markov Random Fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6):707–720, November 1984.

[35] H. Derin and C. S. Won. A Parallel Segmentation Algorithm Using Relaxation with Varying Neighborhoods and its Mapping to Array Procesors. *CVGIP*, 40:54–78, 1987.

[36] P. L. Dobruschin. The Description of a Random Field by Means of Conditional Probabilities and Constructions of its Regularity. *Theory of Probability and its Applications*, XIII(2):197–224, 1968.

[37] R. Duda and P.Hart. *Pattern Classification and Scene Analysis.* Wiley-Interscience, 1973.

[38] O. Faugeras and M. Berthod. Improving Consistency and Reducing Ambiguity in Stochastic Labeling: an Optimization Approach. *IEEE Transactions on PAMI*, 4:412–423, 1981.

[39] T. S. Ferguson. *Mathematical Statistics. A Decision Theoretic Approach.* Probability and Mathematical Statistics. Academic Press, 1967.

[40] K. Fukunaga and T. Flick. Estimation of the Parameters of a Gaussian Mixture Using the Method of Moments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(4):410–416, July 1983.

[41] I. Gaudron and A. Trouvé. Massive Parallelization of Simulated Annealing: An Experimental and Theoretical Approach for Spin-Glass Models. In R. Azencott, editor, *Simulated Annealing: Parallelization Techniques*, pages 163–186. John Wiley & Sons, Inc., 1992.

[42] D. Geiger and F. Girosi. Parallel and deterministic algorithms for MRFs : surface reconstruction and integration. In *Proc. ECCV90*, pages 89–98, Antibes, France, 1990.

[43] D. Geiger and A. Yuille. A Common Framework for Image Segmentation. Technical Report 89-7, Harvard Robotics Lab., 1989.

[44] S. B. Gelfand and S. K. Mitter. On Sampling Methods and Annealing Algorithms. In R. Chellappa, editor, *Markov Random Fields*, pages 499–515. Academic Press, Inc., 1993.

[45] D. Geman. Bayesian image analysis by adaptive annealing. In *Proc. IGARSS'85*, pages 269–277, Amherst, USA, Oct. 1985.

[46] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.

[47] S. Geman, D. Geman, C. Graffigne, and P. Dong. Boundary detection by constrained optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:609–628, 1990.

[48] S. Geman and C. Graffigne. Markov Random Field Image Models and their application to Computer Vision. Research Report, Brown University, 1986.

[49] S. Geman and C. Graffigne. Markov Random Fields Image Models and their Application to Computer Vision. In A. M. Gleason, editor, *Proc. ICM'86*, Providence, 1987. Amer. Math. Soc.

[50] B. Gidas. A Renormalization Group Approach to Image Processing Problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(2):164–180, February 1989.

[51] B. Gidas. Parameter Estimation for Gibbs Distributions from Fully Observed Data. In R. Chellappa, editor, *Markov Random Fields*, pages 471–499. Academic Press Inc., 1993.

[52] R. C. Gonzalez and P. Wintz. *Digital Image Processing*. Addison-Wesley Pub. Co., 1987.

[53] C. Graffigne. A parallel simulated annealing algorithm. Research report, CNRS, Université Paris-Sud, 1984.

[54] C. Graffigne. Parallel Annealing by Periodically Interacting Multiple Searches: An Experimental Study. In R. Azencott, editor, *Simulated Annealing: Parallelization Techniques*, pages 47–80. John Wiley & Sons, Inc., 1992.

[55] C. Graffigne, F. Heitz, F. Prêteux, M. Sigelle, and J. Zerubia. Modèles markoviens hiérarchiques pour l'analyse d'image. Rapport de synthèse GdR 134, Dec. 1994.

[56] C. Graffigne, J. Zerubia, and B. Chalmond. *Segmentation région: approches statistiques*. To appear as a chapter of a book. Masson, 1995.

[57] X. Guyon. *Champs aléatoires sur réseaux: modélisations, statistique et applications*. Masson, 1992.

[58] B. Hajek. A Tutorial Survey of Theory and Applications of Simulated Annealing. In *Proc. 24. Conf. on Decision and Control*, pages 755–760, Lauderdale, FL, December 1985.

[59] B. Hajek. Cooling Schedules for Optimal Annealing. *Mathematics of Operations Research*, 13(2):311–329, May 1988.

[60] F. R. Hansen and H. Elliott. Image Segmentation Using Simple Markov Field Models. *CVGIP*, 20:101–132, 1982.

[61] F. Heitz and C. Kervrann. A statistical model-based approach to unsupervised texture segmentation. In *Proc. SCIA, Norway*, 1993.

[62] F. Heitz, E. Memin, P. Pérez, and P. Bouthemy. A Parallel Multiscale Relaxation Algorithm for Image Sequence Analysis. In *Proc. ICPIP*, Paris, France, June 1991.

[63] F. Heitz, P. Perez, and P. Bouthemy. Multiscale Minimization of Global Energy Functions in Some Visual Recovery Problems. *CVGIP:IU*, 59(1):125–134, 1994.

[64] F. Heitz, P. Perez, E. Memin, and P. Bouthemy. Parallel Visual Motion Analysis Using Multiscale Markov Random Fields. In *Proc. Workshop on Motion*, Princeton, Oct. 1991.

[65] W. D. Hillis. The connection machine. *MIT press*, 1985.

[66] H. P. Hiriyannaiah, G. L. Bilbro, and W. E. Snyder. Restoration of piecewise-constant images by mean-field annealing. *Opt. Soc. Am. A*, 6(12):1901–1912, December 1989.

[67] R. Hummel and S. Zucker. On the foundations of relaxation labeling processes. *IEEE trans. on Pattern Analysis and Machine Intelligence*, 5(3), 1983.

[68] A. H. Jazwinski. *Stochastic Processes and Filtering Theory*, volume 64 of *Mathematics in Science and Engineering*. Academic Press, 1970.

[69] F. C. Jeng and J. M. Woods. Compound Gauss - Markov Random Fields for Image Estimation. *IEEE Trans. Acoust., Speech and Signal Proc.*, ASSP-39:638–697, 1991.

[70] B. Jeon and D. A. Landgrebe. Classification with Spatio-Temporal Interpixel Class Dependency Contexts. *IEEE Trans. on Geoscience and Remote Sensing*, 30(4):663–672, July 1992.

[71] J. M. Jolion and A. Rosenfeld. *A Pyramid Framework for Early Vision*. Series in Engineering and Computer Science. Kluwer Academic Publishers, 1994.

[72] Z. Kato, M. Berthod, and J. Zerubia. A Hierarchical Markov Random Field Model for Image Classification. In *Proc. IMDSP Workshop*, Cannes, France, September 1993.

[73] Z. Kato, M. Berthod, and J. Zerubia. A hierarchical markov random field model and multi-temperature annealing for parallel image classification. Research Report 1938, INRIA, Aug. 1993.

[74] Z. Kato, M. Berthod, and J. Zerubia. Multiscale markov random field models for parallel image classification. In *Proc. ICCV*, Berlin, May 1993.

[75] Z. Kato, M. Berthod, and J. Zerubia. Parallel Image Classification using Multiscale Markov Random Fields. In *Proc. ICASSP*, Minneapolis, Apr. 1993.

[76] Z. Kato, M. Berthod, and J. Zerubia. A Hierarchical Markov Random Field Model and Multi-Temperature Annealing for Parallel Image Classification. *CVGIP:GMIP*, submitted.

[77] Z. Kato, M. Berthod, J. Zerubia, and W. Pieczynski. Unsupervised Adaptive Image Segmentation. In *ICASSP'95*, Detroit, USA, May 1995.

[78] Z. Kato, J. Zerubia, and M. Berthod. Image classification using Markov Random Fields with two new relaxation methods: Deterministic pseudo annealing and modified Metropolis dynamics. Research Report 1606, INRIA, Feb. 1992.

[79] Z. Kato, J. Zerubia, and M. Berthod. Satellite image classification using a modified Metropolis dynamics. In *Proc. ICASSP*, San-Francisco, California, USA, Mar. 1992.

[80] Z. Kato, J. Zerubia, and M. Berthod. Bayesian image classification using markov random fields. In A. M.-D. G. Demoments, editor, *Maximum Entropy and Bayesian Methods*, pages 375–382. Kluwer Academic Publisher, 1993.

[81] R. Kindermann and J. L. Snell. Markov Random fields and their applications. *Amer. Math. Soc.*, 1:1–142, 1980.

[82] S. Kirkpatrick, C. Gellatt, and M. Vecchi. Optimization by simulated annealing. *Science 220, pp 671-680*, 1983.

[83] P. V. Laarhoven and E. Aarts. Simulated annealing : Theory and applications. *Reidel Pub., Dordrecht, Holland*, 1987.

[84] S. Lakshmanan and H. Derin. Simultaneous parameter estimation and segmentation of gibbs random fields using simulated annealing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(8):799–813, Aug. 1989.

[85] S. Lakshmanan and H. Derin. Gaussian Markov Random Fields at Multiple Resolution. In *Markov Random Fields*, pages 131–157. Academic Press, Inc., 1993.

[86] D. A. Langan, K. J. Molnar, J. W. Modestino, and J. Zhang. Use of the Mean-Field Approximation in an EM-Based Approach to Unsupervised Stochastic Model-Based Image Segmentation. In *Proceedings ICASSP'92*, pages III–57–III–60, San Francisco, March 1992.

[87] S. Liu-Yu. *Reconnaissance de formes par vision par ordinateur : application à l'identification de foraminifères planctoniques.* Thèse, Université de Nic,e, Sophia Antipolis, June 1992.

[88] F. Marques, J. Cunillera, and A. Gasull. Hierarchical Segmentation Using Compound Gauss-Markov Random Fields. In *Proc. ICASSP*, San Francisco, Mar. 1992.

[89] J. L. Marroquin. *Probabilistic solution of inverse problems.* PhD thesis, MIT-Artificial Intelligence Lab., 1985.

[90] P. Masson and W. Pieczynski. SEM Algorithm and Unsupervised Statistical Segmentation of Satellite Images. *IEEE Geoscience and Remote Sensing*, 31(3):618–633, May 1993.

[91] E. Memin. *Algorithmes et architectures parallèles pour les approches markoviennes en analyse d'image.* PhD thesis, Université de Rennes I, 1993.

[92] E. Memin, F. Heitz, and F. Charot. Efficient Parallel Non-Linear Multigrid Relaxation Algorithms for Low-Level Vision Applications. *Journal of Parallel Distributed Computing, accepted*, 1994.

[93] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. Equation of state calculations by fast computing machines. *J. of Chem. Physics, Vol. 21, pp 1087-1092*, 1953.

[94] J. Moussouris. Gibbs and Markov Random System with Constrainsts. *Journal of Statistical Physics*, 10(1):11–33, Jan. 1974.

[95] G. K. Nicholls and M. Petrou. A Generalisation of Renormalisation Group Methods for Multi Resolution Image Analysis. In *Proc. ICPR'92*, pages 567–570, 1992.

[96] A. Papoulis. *Probability, Random Variables, and Stochastic Processes.* Series in Systems Science. McGraw-Hill Book Company, 1984.

[97] P. Pérez. *Champs markoviens et analyse multirésolution de l'image: application à l'analyse du mouvement.* PhD thesis, Université de Rennes I, 1993.

[98] P. Perez and F. Heitz. Multiscale Markov Random Fields and Constrained Relaxation in Low Level Image Analysis. In *Proc. ICASSP*, San-Francisco, Mar. 1992.

[99] P. Pérez and F. Heitz. Restriction of Markov Random Fields on Graphs. Application to Multiresolution Image Analysis. Research Report 2170, INRIA-Rennes, March 1994.

[100] W. Pieczynski. Statistical image segmentation. In *Machine Graphics and Vision, GKPO'92*, pages 261–268, Naleczow, Poland, May 1992.

[101] W. Pieczynski. Champs de Markov cachés et estimation conditionnelle itérative. *Traitement du signal*, 11(2), 1994.

[102] J. G. Postaire and C. P. A. Vasseur. An approximate solution to normal mixture identification with application to unsupervised pattern classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 3(2):163–179, March 1981.

[103] W. Pratt. *Digital Image Processing.* Wiley-Interscience, 1978.

[104] S. Rajasekaran. On The Convergence Time of Simulated Annealing. Research Report MS-CIS-90-89, University of Pennsylvania, Department of Computer and Information Science, November 1990.

[105] Y. Rosanov. Markov Random Fields. *Springer Verlag*, 1982.

[106] M. Sigelle, C. Bardinet, and R. Ronfard. Relaxation of Classification Images by a Markov Field Technique - Application to the Geographical Classification of Bretagne Region. In *Proc. European Association of Remote Sensing Lab. Conf.*, Eger, Hungary, Sept 1992.

[107] M. Sigelle and R. Ronfard. Modèles de Potts et relaxation d'images de labels par champs de markov. *Traitement du signal*, 9(6), Mars 1993.

[108] H. L. Tan, S. B. Gelfand, and E. J. Delp. A Cost Minimization Approach to Edge Detection Using Simulated Annealing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(1):3–18, January 1991.

[109] Thinking Machines Corporation, Cambridge, Massachusetts. *Connection Machine Technical Summary*, version 5.1 edition, 1989.

[110] A. Trouvé. Massive Parallelization of Simulated Annealing: A Mathematical Study. In R. Azencott, editor, *Simulated Annealing: Parallelization Techniques*, pages 145–164. John Wiley & Sons, Inc., 1992.

[111] F. Y. Wu. The Potts model. *Reviews of Modern Physics*, 54(1):235–268, January 1982.

[112] L. Younes. Estimation and annealing for Gibbsian fields. *Ann. Inst. Henri Poincaré*, 24(2), 1988.

[113] S. Yu and M. Berthod. A Game Strategy Approach for Image Labeling. *CVGIP – Image Understanding*, Accepted for publication, 1995.

[114] J. Zerubia and R. Chellappa. Mean field approximation using compound Gauss-Markov Random field for edge detection and image restoration. *Proc. ICASSP, Albuquerque, USA*, 1990.

[115] J. Zerubia and R. Chellappa. Mean field annealing using Compound Gauss-Markov Random fields for edge detection and image estimation. *IEEE Trans. on Neural Networks*, 8(4):703–709, July 1993.

[116] J. Zerubia and C. Graffigne. *Segmentation contour: approches statistiques*. To appear as a chapter of a book. Masson, 1995.

[117] J. Zerubia, Z. Kato, and M. Berthod. Multi-Temperature Annealing: A New Approach for the Energy-Minimization of Hierarchical Markov Random Field Models. In *Proc. ICPR'94*, Jerusalem, Israel, Oct. 1994.

[118] J. Zerubia and F. Ployette. Detection de contours et restauration d'image par des algorithmes deterministes de relaxation. Mise en oeuvre sur la machine a connexions CM2. *Traitement du Signal*, Sept. 1991.