1995

Multiresolutional Fault-Tolerant Sensor Integration and Object Recognition in Images.

Lakshman Prasad
Louisiana State University and Agricultural & Mechanical College

INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
313/761-4700 800/521-0600

MULTIRESOLUTIONAL FAULT-TOLERANT SENSOR INTEGRATION AND OBJECT RECOGNITION IN IMAGES

A Dissertation

Submitted to the Graduate Faculty of the
Louisiana State University and
Agricultural and Mechanical College
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

in

The Department of Computer Science

by
Lakshman Prasad
M.S., Indian Institute of Technology, Kanpur, India, 1986
May 1995

Acknowledgements

I would like to thank my advisor Dr. S.S. Iyengar for his consistent support and strong encouragement throughout my doctoral studies. This dissertation was possible in good measure due to his generosity and motivation. Thanks are also due to the members of my doctoral committee, Dr. Don Kraft, Dr. Si-Qing Zheng, Dr. J. Bush Jones, Dr. Leonard Richardson, and Dr. Juan Lorenzo for their time. In particular, I am greatly indebted to Prof. Leonard Richardson for his extremely insightful and elegant exposition of the fundamentals of harmonic analysis which proved to be of great help in my research.

I am grateful to Dr. R.N. Madan of the Office of Naval Research, Arlington, Virginia, for his keen interest in and support of my work in the area of distributed sensor networks. Thanks are due to Dr. George Zweig of the Theoretical Division, Los Alamos National Laboratory, for his kind encouragement and patience during the writing of this dissertation.

I thank my friend Dr. Daryl Thomas for his company and his readiness to help above and beyond the call of duty. I am greatly indebted to my friend Ramana for his invaluable help and innumerable stimulating and fruitful discussions. I would like to express my gratitude to my friends Amit Nanavati and Dr. Vibhas Aravamuthan, for many lively and interesting discussions. They were also extremely generous with their time and effort during the final stages of the writing of this dissertation.

Finally, I would like to acknowledge the love, support and encouragement showered on me by my beloved wife, Gauri, and thank her for tolerating my moods throughout my graduate life.

Contents

List of Tables

List of Figures

Abstract

This dissertation applies multiresolution methods to two important problems in signal analysis. The problem of fault-tolerant sensor integration in distributed sensor networks is addressed, and an efficient multiresolutional algorithm for estimating the sensors' effective output is proposed.

The problem of object/shape recognition in images is addressed in a multiresolutional setting using pyramidal decomposition of images with respect to an orthonormal wavelet basis. A new approach to efficient template matching to detect objects using computational geometric methods is put forward. An efficient paradigm for object recognition is described.

Chapter 1

Introduction

Several structures and phenomena, natural as well as man made have features at several scales in space and time respectively. To describe or understand such structures and phenomena, it is fruitful to analyze and study them at different resolutions. In particular, this is true for signals and functions of one or several dimensions; after all, we use these abstract representations to mathematically capture and characterize structures and phenomena. Multiresolution analysis has been exploited as a valuable and powerful tool in the study of physical and mathematical phenomena for quite some time. Areas of Physics such as fluid dynamics, quantum mechanics, areas in Mathematics such as dynamical systems and fractal geometry have pioneered the use of multiscale analysis. In recent times there has been renewed interest in a systematic development of multiscale methods for analysis of problems. A case in point is the development of wavelet analysis as a powerful framework for studying problems in signal and image processing and numerical analysis, to mention but a few areas.

In this dissertation, two problems are studied using multiscale analysis. The first problem (Chapters 2 – 4) is that of fault-tolerant integration of sensor outputs in Distributed Sensor Networks. This involves the reliable estimation of the sensor outputs of a large battery of sensors measuring a single parameter, amidst occurrence of faults, noise and perturbations among these sensors.

The second problem (Chapters 5 – 11) is that of shape/object recognition in images. This involves searching for the presence or absence of certain (anticipated) objects/shapes in a given image field and recognizing their type, size and position in the image.

While these two problems may appear disparate at a glance, they share some fundamental features. Both involve the isolation and extraction of a subfeature or pattern from a larger and more complex and heterogeneous background. Both problems require simplification and analysis of given compound structure, and as we shall see, both problems are well suited for a multiresolutional approach to solutions.

1.1 Motivation

Apart from a common approach to their solutions another strong reason for addressing these problems together is that both these signal analysis problems are of great importance in several practical applications. Sensor integration problems are routinely encountered in many mundane as well as esoteric situations and applications. For instance several sensors of a simple type may be employed for studying and monitoring a parameter in a power station or grid. A multiplicity of sensors for measuring the same parameter is introduced to compensate for sensor faults and for robustness of the measurement. In another situation, several temperature and radiation gauges may be distributed spatially inside a nuclear reactor, to monitor heat transfer and radiation imbalances. Thus, sensor integration is an important aspect of many control systems.

Object and shape recognition is of great value to machine and robotic vision. It is crucial to automation of industrial processes as well as vital military and intelligence applications such as reconnaissance, remote surveillance, target identification etc. Indeed, the importance of both problems in today's world cannot be overstated.

1.2 Organization

This dissertation is composed of two parts. The first part addresses the problem of signal integration in distributed sensor networks. The second part proposes a efficient multiresolutional template matching paradigm for object recognition.

Chapters 2 through 4 discuss the problem of fault tolerant-sensor integration in distributed sensor networks, and put forward an efficient algorithm for integrating the signal estimates from a network of sensors in a fault-tolerant manner.

Chapter 5 introduces the problem of object recognition with a discussion of prior research in this area. Chapters 6 and 7 briefly introduce wavelet analysis in the context of orthonormal pyramidal decomposition of images.

Chapters 8 through 10 describe a general and efficient paradigm for object recognition in a pyramidal scheme using computational geometric methods to reduce the computational overhead involved in template matching with a large template database.

Chapter 11 presents a short summary of the merits and contributions of the object recognition paradigm.

Chapter 2

Fault-Tolerant Sensor Integration Using Multiresolution Decomposition

2.1 Introduction

Signal integration has been shown to have wide ranging applications in areas such as radar tracking and target detection. This includes the problem of fault-tolerant integration of information from multiple sensors, mapping and modeling the environment space and task level complexity issues of the computational model. Further, these techniques have to be robust in the sense that even if some of the sensors are faulty, the integrated output should still be as reliable as possible.

In this chapter, a new approach to the problem of sensor integration using techniques of *multiresolution analysis* is discussed. Multiresolution analysis may be described as signal analysis in frequency channels of constant bandwidth on a logarithmic scale.

The cumulative signal from all the sensors is analyzed at various resolutions, starting from a coarse resolution and proceeding to successively finer scales. In a coarse-to-fine strategy, a minimum of detail necessary for recognition is processed. The approximation of a signal f at a resolution r is defined as an estimate of f derived by uniformly sampling f, r times per unit length. Tanimoto and Pavlidis [24] have developed efficient algorithms to compute the approximation of a function at different resolutions.

In what follows, the idea of multiresolution analysis is applied to the problem of fault-tolerant integration of abstract sensor estimates when the number of sensors is very large and a large number of sensor faults are *tame*. The key idea is the construction of a simple function from the outputs of all the sensors and resolution of this function at various scales to isolate the region over which the correct sensors lie. An optimal algorithm which implements this idea efficiently is described.

2.2 The Sensor Integration Problem

A Distributed Sensor Network consists of spatially distributed sensors that detect and quantify a certain phenomenon via its changing parameters. These readings are sent at regular intervals of time to processing units that integrate the readings from clusters of sensors and give outputs whose form is much the same as the inputs of the individual sensors. Output from processors representing clusters of sensors are later integrated to get a complete picture of the spatially distributed phenomenon. However, before integration is performed at the processor level, it is necessary to have reliable estimates at each processor. Each sensor in a cluster measures the same parameter. It is possible that some of these sensors are faulty. Hence it is desirable to make use of this redundancy of the readings in the cluster to obtain a correct estimate of the parameters being observed. In short, a fault-tolerant technique of sensor integration is sought.

Marzullo [26] has addressed the problem of fault-tolerant integration of abstract interval estimates and has generalized his estimates to multidimensional sensors [27]. An illustration of their method for one dimensional sensors is given in Figure 2.1.

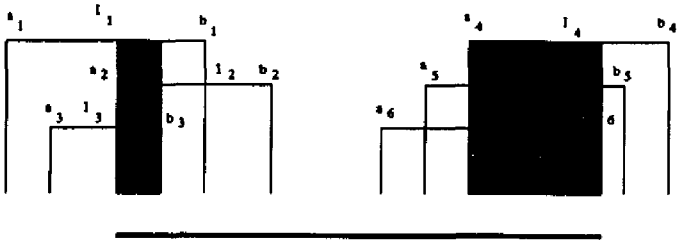


Figure 2.1: Integration of Interval Estimates

$$a_1 < a_3 < a_2 < b_3 < b_1 < b_2 < a_6 < a_5 < a_4 < b_6 < b_5 < b_4$$

Prasad et al [28] have proposed a method of obtaining sensor estimates with high reliability by considering the problem when the number of sensors is large, and most of the faulty sensors are tamely faulty. This method has also been generalized to multidimensional sensors [29].

The technique developed in [28] and [29] is a polling technique, which computes the intersections of sensor outputs and the associated reliability measures. Since the number of these intersections is very large, the method is not useful for real-time applications.

In order to obtain a method of fault-tolerant sensor integration that is more feasible for real-time applications, a function called the *overlap function* is introduced [28]. The multiresolution analysis approach to one-dimensional sensors discussed below generalizes easily to higher dimensions.

2.3 Preliminaries

The relevant definitions from [28] are reviewed here for completeness.

Definition 2.1: An *Abstract Sensor* is a sensor that reads a physical parameter and gives out an abstract interval-estimate I_s which is a bounded and connected subset of the real line $\mathbf{R}$.

Definition 2.2: A *Correct Sensor* is an abstract sensor where the interval estimate contains the actual value of the parameter being measured. If the interval estimate does not contain the actual value of the parameter being measured, it is called a *Faulty sensor.*

Definition 2.3: Let sensors $S_1, ..., S_N$ feed into a processor P. Let the abstract interval estimate of S_j be $I_j; (1 \leq j \leq N)$, the closed interval $[a_i, b_i]$ with endpoints a and b. Define the *Characteristic Function* χ_j of the j-th sensor $S_j, 1 \leq j \leq N$ as follows:

$$\chi_j : \mathbf{R} \rightarrow \{0,1\} \qquad \chi_j\,(x) = \begin{cases} 1 \;\; \forall\, x \in I_j \\ 0 \;\; \forall\, x \notin I_j \end{cases} \quad \forall\, 1 \leq j \leq N \tag{2.1}$$

Definition 2.4: Let $O(x) = \sum_{j=1}^{N} \chi_j(x)$ be the 'overlap function' of the N abstract sensors. For each $x \in \mathbf{R}$, $O(x)$ gives the number of sensor intervals in which x lies; that is, the number of intervals overlapping at x.

Definition 2.5: A sensor is *tamely faulty* if it is a faulty sensor and if its output overlaps with that of a correct sensor.

2.4 Comments on Tame Faults

If we tolerate at most f faults among N sensors, then by taking all $(N - f)$ intersections of the N sensor interval estimates, we are assured that the correct value of the parameter lies in one of these intersections. Marzullo [26] computes the integrated output as the smallest connected interval containing all the $(N - f)$

intersections. However, when the number of sensors is large and the number of faults cannot be strictly bounded, the $(N - f)$ intersections tend to be scattered wildly over the real line, giving poor output estimates. In order to improve the output estimate in these cases, we must be able to further evaluate the $(N - f)$ intersections to choose the 'best possible' intersection which contains the correct value with high reliability.

In the method proposed here it is assumed, as before [28], that the number of sensors is very large, that most faults are tame, and that there is no bound on the number of faults.

As the sensors are sampled synchronously at various time intervals, we order the sensors *a priori* by labeling them, dynamically maintain their overlap function $O(x)$, and analyze it at various scales to obtain successively smaller regions which contain the correct value of the parameter observed.

The function $O(x)$ is the sum of the characteristic functions of the abstract interval estimates. The value of $O(x)$ at any point x is the number of intervals overlapping at the point x. The structure of the function is fairly simple (see fig. 2.2.)

Since there are finitely many sensors and each sensor is represented by an abstract interval estimate of bounded length, $O(x)$ has compact support. By definition, $O(x)$ is a non-negative function. It has several 'crests' in its profile representing regions of maximal overlap of intervals (fig. 2.2.)

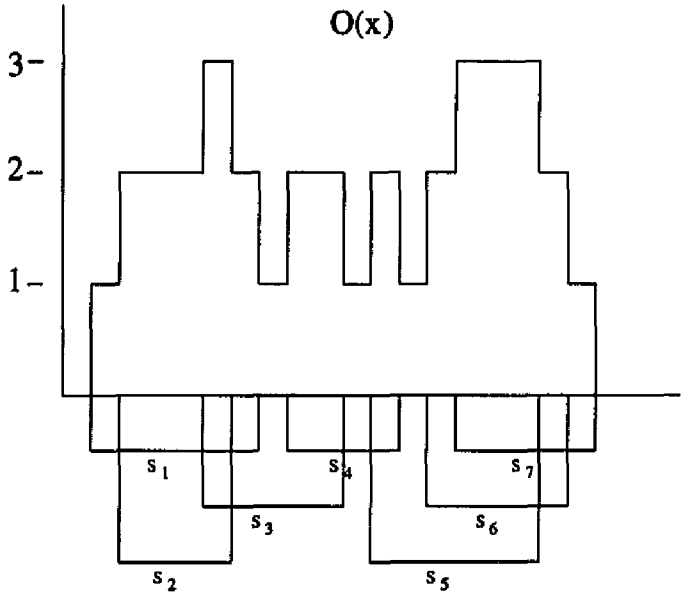


Figure 2.2: The Overlap Function O(x) for a set of 7 Sensors

Chapter 3

Multiresolution Decomposition

3.1 Introduction

Given a sequence of increasing resolutions $\{r_j\}_{j\in Z}$, the details of a function $f(x)$ at the resolution r_j are defined as the difference of information between the approximations of $f(x)$ at the resolution $r_j + 1$ and the approximation at the resolution r_j. A *Multiresolutional* representation also provides a simple hierarchical framework for interpreting the signal content. For instance, it is hard to recognize and identify a local feature inside an image, without any information of the global context in which it is embedded. It is therefore natural to first analyze image details at a coarse resolution and then increase the resolution.

The approximation of a signal $f(x)$ at a resolution r is defined as an estimate of $f(x)$ derived from r measurements per unit length. These measurements are computed by uniformly sampling at a rate r the function $f(x)$ smoothed by a low-pass filter whose bandwidth is proportional to r. In order to be consistent when the resolution varies, these low-pass filters are derived from a unique scaling function which is dilated by the resolution factor r.

The overlap function $O(x)$ is analyzed multiresolutionally for a robust peak corresponding to the region of overlap of correct sensors, as follows: Starting at the coarsest resolution, we select those crests with the highest peaks (wavelet components with the largest amplitude) and choose the crest with the widest spread.

At the next higher resolution, this crest is again inspected for crests within it with highest amplitudes and among these crests, the one with the widest spread is retained for similar analysis at the next resolution. This procedure results in isolating those regions of the real line over which $O(x)$ has a maximum value, corresponding to high overlap degree. Figures 3.3, 3.4, 3.5, 3.6 and 3.7 illustrate this procedure. The advantage of this method is significant from the point of view of computational speed, since the coarse-to-fine processing leads to elimination of large regions of the support of $O(x)$ at each resolution.

This heuristic is rigorously formulated in the next section in order to obtain a real time algorithm that dynamically maintains $O(x)$ and obtains the narrowed output estimate. The maintenance of $O(x)$ requires $O(NlogN)$ time where N is the number of sensors. This follows directly from the fact that sensor intervals need to be sorted first, according to their beginning and end points; a process that has a computational complexity of $\Omega(NlogN)$. It is then verified that this method leads to results comparable to our earlier results and those of Marzullo, by simulating sensor failures and using the algorithm to obtain sensor estimates.

In the model of abstract sensors employed here, it is assumed that i) A large number of sensor faults are tame, and that ii) the length of each interval estimate is bounded below by l, and above by L, where $l < L$ and l, L are positive real numbers.

Figure 3.1 describes the regions about the correct parameter value c where the faulty sensors cluster. A very large interval estimate is too inaccurate to be of any value and hence may be discarded. On the other hand, a very small interval estimate would not be amenable for fault-tolerance analysis. A minimum tolerance of $\pm l/2$ is built into the abstract sensors, and so we may assume that the width of

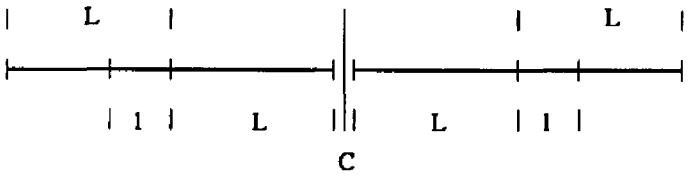


Figure 3.1: The regions about the correct parameter value c where the faulty sensors cluster has $L = 2l$

each interval is at least l. These two assumptions imply that the tame faults cluster in a bounded neighborhood around the correct value of the measured parameter. When the number of faulty sensors are significant, since most faults are tame, this results in overlaps of the faulty sensors amongst themselves and boosts the value of $O(x)$ in the neighborhood of the correct value of the parameter, thus reinforcing the $(N - f)$ intersection containing the correct value.

Let T be the number of tamely faulty sensors. These may range in width from l to L. A tamely faulty sensor must intersect with a correct sensor. Therefore its endpoint nearest to the correct value c must lie within a distance of at most L from c. Thus at most $(1 + \lfloor L/l \rfloor)$ tamely faulty sensors can be accommodated on either side of c with no two of them overlapping. That is, at least $\lceil T/2(1+\lfloor L/l \rfloor)\rceil$ tamely faulty sensors overlap over a region of width at least l within a distance of at most $2L$ from c.

When the number of intersections of tamely faulty sensors is $\lceil T/2(1+\lfloor L/l \rfloor)\rceil$, the width of this intersection is actually at least $2(l + L)$. When the number of intersections is T, then this results in a peak with spread of at least l. This clustering reinforces the width and height of the 'correct' $(N - f)$ intersection by adding in its neighborhood a peak of area Tl at least. In general, this results in a taller and wider peak in the neighborhood of c. The wildly faulty sensors, on the other hand, are random in their location on the real line and being uncorrelated,

tend not to cluster in any small neighborhood. Thus the $(N - f)$ intersections resulting from them have shorter and narrower peaks representing them in $O(x)$.

3.2 Multiresolution of the Overlap Function

If S_i $(1 \leq i \leq N)$ are N abstract sensors with their interval estimates $[a_i, b_i]$ $(1 \leq i \leq N)$ having characteristic function χ_i $(1 \leq i \leq N)$ such that

$$\chi_i(x) = \begin{cases} 1 \;\; if \; x \in [a_i, b_i] \\ 0 \;\; if \; x \notin [a_i, b_i] \end{cases}, \tag{3.1}$$

then the overlap function $O(x)$ of these N sensors is given by $O(x) = \sum_{i=1}^{N} \chi_i(x)$.

For each j, $O(x)$ can be sampled at regular intervals $1/2^j$ to obtain the j-th resolution of $O(x)$ at scale $1/2^j$ as a linear combination of a set of functions obtained by scaling and translating a single function.

Let

$$\sigma(x) = \begin{cases} 1 \;\; if \; 0 \leq x \leq 1 \\ 0 \;\; \text{otherwise} \end{cases}; \tag{3.2}$$

Let $\alpha \in R$ and $j \in Z$. Consider the functions

$$\left\{\sigma(2^j(x-\alpha) - n\right\}_{n=-\infty}^{\infty} \sigma(2^j(x-\alpha)-n) = \begin{cases} 1 \;\; if \; \alpha + n/2^j \leq x < \alpha + (n+1)/2^j \\ 0 \;\; \text{otherwise} \end{cases}. \tag{3.3}$$

Without loss of generality, we may assume $0 \leq \alpha < 1/2^j$. Note that

$$[\alpha + n/2^j, \; \alpha + (n+1)/2^j) \bigcap [\alpha + (n+1)/2^j, \; \alpha + (n+2)/2^j) = \emptyset \tag{3.4}$$

and

$$\bigcup_{N=-\infty}^{\infty} [\alpha + N/2^j, \alpha + (N+1)/2^j] = \mathbf{R} \tag{3.5}$$

The j-th resolution of $O(x)$ with respect to the functions $\{\sigma(2^j(x-\alpha)-n\}_{n=-\infty}^{\infty}$ is denoted by $O_\alpha^j(x)$, and is given by

$$O_\alpha^j(x) = \sum_{n=-\infty}^{\infty} O(\alpha + 2^{-j})\sigma(2^j(x-\alpha)-n) \tag{3.6}$$

Since $O(x)$ has compact support, the above summation is actually over finitely many n. If the interval estimates of the sensors S_i are $[a_i, b_i]$ $(1 \leq i \leq N)$ and

$a = min_{1\leq i\leq N}\{a_i\}$ and $b = max_{1\leq i\leq N}\{b_i\}$, then

$$O_\alpha^j(x) = \sum_{n=\lceil 2^j(a-\alpha)\rceil}^{\lceil 2^j(b-\alpha)\rceil} O(\alpha + n2^{-j})\sigma(2^j(x-\alpha)-n) \tag{3.7}$$

Thus $O_\alpha^j(x)$ is obtained from $O(x)$ by sampling $O(x)$ at the points $\{\alpha+n2^{-j}\}_n$. O_α^j is a function whose features are of size $1/2^j$ or greater. To study the effect of sampling in the above manner it is sufficient to study the sampling of the characteristic function of an arbitrary interval $[a, b]$, since $O(x)$ is a linear combination of characteristic functions.

Consider the test function

$$g(x) = \begin{cases} 1 \;\; if \; x \in [a,b] \\ 0 \;\; if \; x \notin [a,b] \end{cases} \tag{3.8}$$

Case i) $b - a \geq 1/2^j$, i.e. $(g(x)$ is a feature bigger than the scale width. $a \in [\alpha + n/2^j, \alpha + (n+1)/2^j)$ for some $n, b \in [\alpha + m/2^j, \alpha + (m+1)/2^j)$, and $n < m$. Thus

$$g_\alpha^j(x) = \sum_{r=-\infty}^{\infty} g(\alpha + r2^{-j})\sigma(2^j(x-\alpha)-r) =$$

$$g(\alpha + n2^{-j})\sigma(2^j(x-\alpha)-n) + \ldots + g(\alpha + m2^{-j})\sigma(2^j(x-\alpha)-m) \; if \; a = \alpha + n/2^j$$

or

$$g(\alpha+(n+1)2^{-j})\sigma(2^j(x-\alpha)-n-1)+\ldots+g(\alpha+m2^{-j})\sigma(2^j(x-\alpha)-m) \; if \; a > \alpha+n/2^j$$

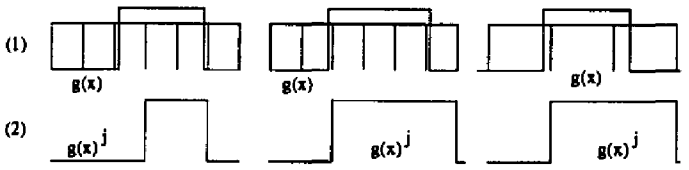


Figure 3.2: Computational Characterization of Support of $g(x)$

Therefore we have

$$g_\alpha^j(x) = \begin{cases} 1 \;\; if\; \alpha + n/2^j \leq x < \alpha + (m+1)/2^j \\ 0 \;\; \text{otherwise} \end{cases} \qquad if\; a = \alpha + n/2^j \qquad (3.9)$$

and

$$g_\alpha^j(x) = \begin{cases} 1 \;\; if\; \alpha + (n+1)/2^j \leq x < \alpha + (m+1)/2^j \\ 0 \;\; \text{otherwise} \end{cases} \qquad if\; a > \alpha + n/2^j \; (3.10)$$

Thus there are two things which may happen independently to the support of $g(x)$:

i) it may shrink on the left by at most $1/2^j$ and

ii) it may extend on the right by at most $1/2^j$ (see 3.2).

It will be seen later that one has to correct for a positive shrinkage of the support of a feature, so as not to lose any information (correct value of the parameter measured). This is done by resolving over a region bigger than the one at hand by $1/2^j$ on the left. The extension ('smearing') of support will decrease with further resolution, and does not pose a problem with regard to accuracy of the estimated output.

Case ii) $b - a < 1/2^j$. ($g(x)$ is a feature smaller than the scale width). If $a, b \in [\alpha + n/2^j, \alpha + (n+1)/2^j)$ for some n, then $g_\alpha^j(x) = 0 \;\forall\; x$ i.e., the feature will not appear at scale $1/2^j$. If $a \leq 1/2^j \leq b$ for some n then

$$g_\alpha^j(x) = \begin{cases} 1 \;\; if\; \alpha + n/2^j \leq x < \alpha + (n+1)/2^j \\ 0 \;\; \text{otherwise} \end{cases} \qquad (3.11)$$

that is, $g(x)$ will appear as a feature of size $1/2^j$ shifted to the right by at most $b-a$. This will diminish in size with further resolution, and $g(x)$ will be recovered by correction to the left and resolution.

We note that changing α will not produce any advantage insofar as sampling $O(x)$ is concerned since location of the sampling points $\{\alpha + n/2^j\}$ with respect to $O(x)$ is arbitrary. We may thus conveniently let $\alpha = 0$ and henceforth sample $O(x)$ at points $\{n/2^j\}$ to obtain the j-th resolution. Thus

$$O^j(x) = \sum_{n=\lceil 2^j a \rceil}^{\lceil 2^j b \rceil} O(n2^{-j})\sigma(2^j x - n) \quad j \in \mathbf{Z} \tag{3.12}$$

While considering resolution of $O(x)$, it is important to choose the scale appropriately. Too large a scale will provide no useful information about the structure of O, while too small a scale would not isolate the features important to us, by bringing in unnecessary detail. Further, since each sensor has width at least l, it is desirable to start off with a scale smaller than (or the same order as) l i.e., choose $j = \lceil log(1/l) \rceil$. Thus each sensor will figure as a feature at least as big as $1/2^j$.

The fluctuations in $O(x)$ occur at the points a_i, b_i $(1 \leq i \leq N)$ which are the end points of the interval estimates. If a is the least of the a_i and b is the largest of the b_i, then the average number of fluctuations per unit length is given by $2N/(b-a)$. So in order to capture all the fluctuations we would have to resolve at least to a level $j > log(2N/(b-a))$.

3.3 Selection of Robust Peaks

At the j-th level of resolution O_α^j can be looked upon as a series of juxtaposed peaks. In other words, consider the sequence $\{O(n/2^j)\}$. This sequence is a

concatenation of several bitonic sequences, each of which increases first and then decreases (for details about bitonic sequences, see [30]). Each bitonic sequence which increases first and then decreases corresponds to a peak in O^j_α. We wish to isolate those peaks which are the tallest and have the widest spread, for it is in the region over which these peaks lie that the correct value of the parameter being measured is most likely to be found. Since the characteristic function of each sensor adds an area numerically equal to the sensor's width to the area under $O(x)$, a good measure of the robustness of a peak is the area under it.

At the j-th resolution consider the sequence $\{O(n/2^j)\}_n$. This is a finite sequence since the support of O is finite. Let there be p peaks (or p bitonic sequences) in O^j . Thus the sequence $\{O(n/2^j)\}$ can be rewritten as

$$\left\{O\left(\frac{n_0}{2^j}\right), O\left(\frac{n_0+1}{2^j}\right), O\left(\frac{n_1}{2^j}\right), \ldots, O\left(\frac{n_{p-1}+1}{2^j}\right), O\left(\frac{n_p}{2^j}\right)\right\}, \tag{3.13}$$

where the subsequence $\{O((n_{k-1}+1/)2^j), \ldots, O(n_k/2^j)\}$ is the k-th bitonic sequence from the left. Therefore the area under this peak is given by $1/2^j \sum_{n=n_{k-1}}^{n_k} O(n/2^j)$. Since the factor $1/2^j$ is common to the areas of all peaks at the j-th resolution, we may make the area 'scale-free' by dropping this factor and writing the area of the k-th peak at level j as

$$\mathcal{A}^j(k) = \sum_{n=n_{k-1}}^{n_k} O(n/2^j). \tag{3.14}$$

We then select the peak with the largest area and ignore the other peaks. The function O is further resolved over the regions over which these largest peaks occur, and the process is repeated until a satisfactory region of the real line is isolated as the most likely candidate for containing the correct value of the parameter being measured by the sensors. However, before resolving a certain selected peak further, we correct the region over which the resolution is to be carried out by adding a

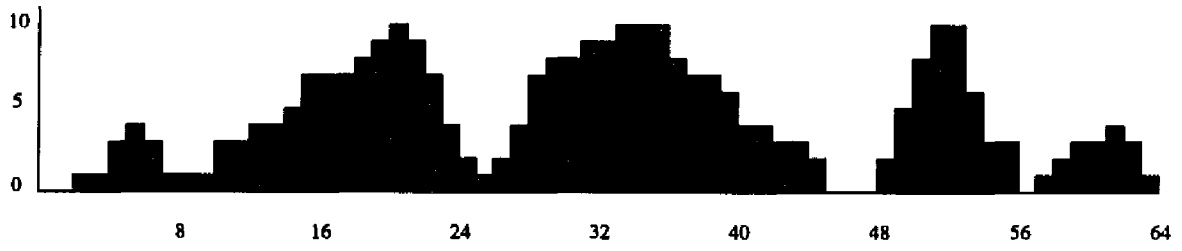


Figure 3.3: $O(x)$, shaded region indicates portion to be resolved

Figure 3.4: $O^{-3}(x)$, shaded region & region of width 8 at left to be resolved

segment of length $1/2^j$. Figures 3.3, 3.4, 3.5, 3.6 and 3.7 show the property of the coarse-to-fine scheme for isolating robust peaks.

If at the j-th resolution the k-th peak is selected as the peak with the largest area under it, then the region over which the resolution of $O(x)$ is performed again is $[(n_{k-1}+1)/2^j, n_k/2^j]$ with a correction of length $1/2^j$ at the left.

Therefore, $O(x)$ is resolved over $[n_{k-1}/2^j, n_k/2^j]$. This process is continued until the interval to be resolved is smaller than the maximum acceptable width for output estimate, or when further resolution does not reduce interval width. The final corrected region with the largest value of O over it is accepted as the final output estimate.

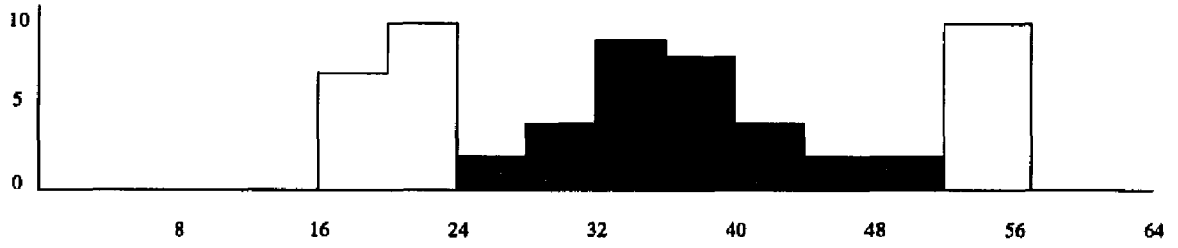


Figure 3.5: $O^{-2}(x)$, shaded region & region of width 4 at left to be resolved

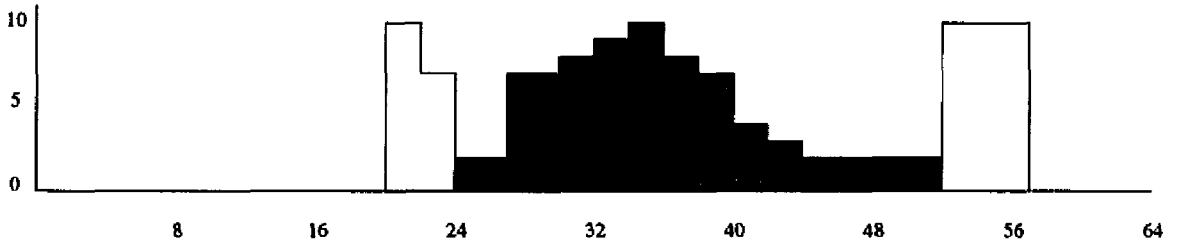


Figure 3.6: $O^{-1}(x)$, shaded region & region of width 2 at left to be resolved

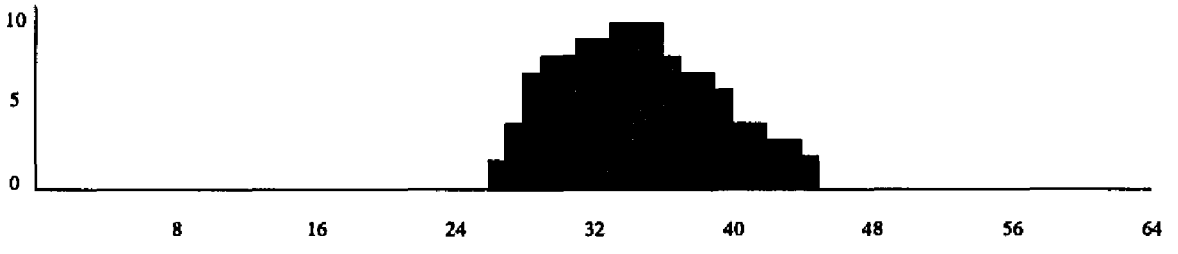


Figure 3.7: $O^{0}(x)$, shaded region & region of width 1 at left to be resolved over. At this point, further resolution is stopped and the region of peak value is selected

3.4 The Multiresolution Algorithm

The inputs to the algorithm are the end points a_i, b_i of the interval estimate $[a_i, b_i]$ of the sensors $S_i, 1 \leq i \leq N$, and the lower and upper bounds of resolution $j_0 = \lceil log(1/l) \rceil$ and $j_1 > \lceil log(2N/Supp(O(x))) \rceil$. The algorithm is shown in the figure overleaf.

Procedure **RESOLVE**, which resolves $O(x)$ to obtain an approximation of O at the j-th resolution over a given interval, is given below. It yields the indices of two points at the j-th resolution, over which the largest or most prominent peak occurs, at the j-th resolution.

Figures 3.3, 3.4, 3.5, 3.6 and 3.7 indicate the various stages during the execution of this algorithm graphically.

Procedure **RESOLVE** involves only scanning and hence is linear in the number of sensors N. Since the average density of fluctuation in O is $2N/Supp(O(x))$, the level of resolution j required to capture almost all the measures of O is given by $j > log(2N/Supp(O(x)))$.

If we assume that the parameter being measured by the sensors is known to lie between certain bounds, then $j > logN + C$ where C is some constant. Thus procedure **RESOLVE** will be called, on an average, $O(logN)$ times. Hence the average computational complexity of the algorithm is $O(NlogN)$.

begin

1. Construct the array of ordered pairs:

$$[(a_1,1),(b_1,-1),(a_2,1),(b_2,-1),\ldots,(a_N,1),(b_N,-1)]$$

2. Sort this array in increasing order with respect to the first component of the ordered pairs to obtain the array $[(\alpha_1,\sigma_1),(\alpha_2,\sigma_2),\ldots,(\alpha_{2N},\sigma_{2N})]$, where each α_i is some a_j or $b_j, \alpha_i, \leq \alpha_{i+1} \quad 1 \leq i \leq 2N$, and

$$\sigma_i = \begin{cases} 1 & \text{if } \alpha_i \text{ is an } a_j \\ 0 & \text{if } \alpha_i \text{ is a } b_j \end{cases}.$$

3. Construct the array $[(-\infty,0),(\alpha_1,\sigma_1),\ldots,(\alpha_{2N},\sum_{j=1}^{2N}\sigma_j),(\infty,0)]$ representing the overlap function $O(x)$. Note that $\sigma_1 = +1, \sum_{j=1}^{i}\sigma_j = 0$, and $O(x) = \sum_{j=1}^{i}\sigma_j \ \forall\ x\ \alpha_i \leq x < \alpha_{i+1}\ ,\ 0 \leq i \leq 2N$ where $\alpha_0 = -\infty$ and $\alpha_{2N+1} = \infty$). Set $n_{j_0} = \lfloor 2^{j_0}\alpha_1 \rfloor$ and $n'_{j_0} = \lceil 2^{j_0}\alpha_{2N} \rceil$

4. **while** $j_0 \leq i < j_1$ **do** $\quad RESOLVE[i; n_{i-1}, n'_{i-1}]; \quad i \leftarrow i+1$; **enddo**
Sample $O(x)$ between $n_{j_1}/2^{j_1}$ and $n'_{j_1}/2^{j_1}$ to obtain $O^{j_1}(x)$, the approximation of $O(x)$ at the j-th resolution. Choose that subinterval of $[n_{j_1}/2^{j_1}, n'_{j_1}/2^{j_1}]$ over which $O^{j_1}(x)$ attains a maximum (or takes values greater than a specified value) and accept this subinterval as the integrated output estimate of the N sensor estimates

end.

Figure 3.8: The Multiresolution Algorithm

procedure $RESOLVE[j; n_{j-1}, n'_{j-1}]$

begin

1. Resolve $O(x)$ at scale 2^{-j} by sampling it over the interval $[(n_{j-1} - 1)/2^{j-1}, n'_{j-1}/2^{j-1}]$ at the points $(2n_{j-1} - 2)/2^j, (2n_{j-1} - 1)/2^j, \ldots, 2n'_{j-1}/2^j$ to obtain $O^j(x)$, the approximation of $O(x)$ at the j-th resolution, represented by the array

$$[(-\infty, 0), \ldots, (n/2^j, \sum_{j=1}^{k_n} \sigma_j), \ldots, (\infty, 0)], \ (2n_{j-1} - 2 \leq n \leq 2n'_{j-1})$$

and $O^j(x) = \sum_{j=1}^{k_n} \sigma_j$ where $n/2^j \leq x \leq (n+1)/2^j$ and $(n/2^j - \alpha_{k_n}) < 1/2^j$

2. Choose n_j and n'_j, where $n_j < n'_j$ and $2n_{j-1} - 2 \leq n_j \leq 2n'_{j-1}$ such that $\{O(n/2^j\}_{n=n_j}^{n'_j}$ is a contiguous bitonic subsequence of $\{O(n/2^j)\}_{n=2n_{j-1}-2}^{2n'_{j-1}}$, which first increases and then decreases, and which has the largest sum i.e., the subsequence with the maximum sum among all such bitonic subsequences

end.

Figure 3.9: The RESOLVE Procedure used by the Multiresolution Algorithm

Chapter 4

Experimental Validation

4.1 Introduction

In this chapter the algorithm described in section 3.4 is implemented, and a parameter-driven simulator to evaluate the performance of the algorithm is developed. Briefly, the simulator parses the input parameters which include the number of sensors, the numbers of tamely faulty and wildly faulty sensors, the upper and lower limits for the sensor interval widths, and the correct value. It then constructs randomly generated intervals for the sensors according to the scheme described in the following section. The overlap function is computed from a sorted list of the extreme points of all the sensors.

The overlap function is then sampled at increasing levels of resolution with the sampling frequency gradually increasing. The algorithm terminates when only one peak remains or when an arbitrarily fixed resolution is reached.

The overall complexity of the simulator is dominated by the sorting operation that needs to be performed over the endpoints of the sensor intervals. Thus, the computational complexity of the simulator is $O(Nlog(N))$ where N is the total number of interval endpoints. This complexity is optimal in the sense that sorting $O(N)$ values cannot be completed in less than $O(Nlog(N)$ time.

4.2 Construction of Sensor Intervals

The correct valued sensors (these intervals contain the correct physical value being estimated) are constructed as

$$s_i = L - \omega.\eta; \quad e_i = s_i + \omega \tag{4.1}$$

where s_i is the start-point of the interval, and e_i is the end-point of the interval. ω is the width of the sensor interval estimate and is constrained to lie between ω_{min} and ω_{max}. L in the equation above is the correct value of the physical quantity estimated. η is a random number uniformly distributed over $(0.0, 1.0]$. For different sensors, ω is a random number uniformly distributed over $[\omega_{min}, \omega_{max}]$. The intervals for tamely faulty sensors are constructed as

$$s_i = L + t_{off}.\eta; \quad e_i = s_i + \omega. \tag{4.2}$$

where t_{off}, a parameter of tame faults, denotes the critical distance from the correct value within which one of the end points of each tamely faulty sensor interval must lie [1]. Intervals for wildly faulty sensors are constructed as

$$s_i = L + t_{off} + \eta\ (L_{max} - t_{off} - \omega); \quad e_i = s_i + \omega \tag{4.3}$$

where L_{max} is the maximum value any sensor can estimate. Faulty sensor intervals (both wild and tame) are located randomly on both sides of the correct value L.

4.3 Computation and Sampling of the Overlap Function

The overlap function is described in detail in section 2.4. The array

$$[(-\infty, 0), (\alpha_1, \sigma_1), ..., (\alpha_i, \sum_{j=1}^{i} \sigma_j), ..., (\alpha_{2N}, \sum_{j=1}^{2N} \sigma_j), (\infty, 0)]$$

[1] both end-points of wildly faulty sensor intervals lie at a distance of at least t_{off} from L

represents the overlap function $O(x)$. In the above equation, α_i is an extreme point of some interval (either s_j or e_j for some j) such that $\alpha_i \leq \alpha_{i+1} \quad 1 \leq i \leq 2N$ and σ_i takes the value one if α_i is a start-point of some interval and takes the value negative one if α_i is an end-point of some interval.

The overlap function is sampled at increasing sampling frequencies as described in section 3.2. The simulation begins with the interval whose end-points are $n_{j_0} = 2^{j_0}\alpha_1$ and $n'_{j_0} = 2^{j_0}\alpha_{2N}$. These end-points are replaced by the end-points of the peak with the largest area under it, found at the current sampling rate for resolution at the next higher level of resolution. The simulation then continues until no further refinement is possible or until a user-specified resolution is reached.

Figures 4.1 through 4.10 show the results of the application of our algorithm to several instances of the sensor interval integration problem. The overlap function appears as a jagged, piecewise constant curve. The vertical center line in the middle of each figure is the correct physical value being estimated by the sensors. The sampled overlap function is shown by the boxy curves. The peak of the sampled overlap function with the largest area under it, is shown delimited by the leftmost and rightmost vertical lines. These lines move closer and closer at finer scales of resolution as seen in the figures. The actual sensor intervals are also shown in some figures overlaid with the overlap function; these appear as horizontal line segments, where shown.

4.4 Simulation Results

Tables 4.4 through 4.4 show the regions of interest in the overlap function $O(x)$ as the sampling frequency is varied from coarse to fine. When a rise or fall of the

Table 4.1: 200 sensors; 100 wildly faulty, 50 tamely faulty

Sampling Frequency	Resolution Level	Sensor Interval		
		Start Point	End Point	Width
coarse	3	-0.449277	0.673841	1.123118
medium	4	-0.191211	0.128103	0.319314
medium	5	-0.191211	0.128103	0.319314
medium	6	-0.191211	0.128103	0.319314
fine	7	-0.022714	0.003010	0.025724

Table 4.2: 100 sensors; 50 wildly faulty, 30 tamely faulty

Sampling Frequency	Resolution Level	Sensor Interval		
		Start Point	End Point	Width
coarse	3	-0.675956	0.663665	1.339621
medium	4	-0.675956	0.663665	1.339621
medium	5	-0.115485	0.074100	0.189585
fine	6	-0.115485	0.074100	0.189585

sampled function is so sharp that it is completely missed at the current resolution level, the region of interest at the previous level of resolution is itself used at the next level. For example, in Table 4.4, the three rows marked *medium* under sampling frequency exhibit this behavior.

Tables 4.4 and 4.4 both show the intermediate stages of resolution for 100 sensors but with different proportions of faulty and correct sensors. It is clear from these tables that as the proportion of correct sensors increases, the central peak (containing the correct physical value — see figures 4.2, 4.3 & 4.4) sustains at different levels of resolution.

Figure 4.2 and 4.3 graphically shows the information summarized in Table 4.4. As seen from the figure and the table, there are only two broad scales in the overlap

Table 4.3: 100 sensors; 10 wildly faulty, 10 tamely faulty

Sampling Frequency	Resolution Level	Sensor Interval		
		Start Point	End Point	Width
coarse	3	-0.627078	0.379362	1.006440
medium	4	-0.627078	0.379362	1.006440
medium	5	-0.627078	0.379362	1.006440
fine	6	-0.627078	0.379362	1.006440

function — a coarse scale (first two rows of table) and a fine scale. The correct sensor intervals all contain the correct value — the vertical center line. The tamely faulty sensor intervals are clustered at the top of the two figures located not too far from the correct value; the wildly faulty sensor intervals are located on either side of the correct value a significant distance away from it.

As an illustration of the extreme case behavior of our algorithm, we include two figures (4.4 and 4.5 through 4.7) Figure 4.4 shows the overlap function corresponding to 100 sensors; 10 wildly faulty and 10 tamely faulty (see Table 4.4 for data). Notice that the most significant peak is sustained at every intermediate level of resolution — a phenomenon that faithfully repeats whenever the fraction of correct valued sensors among all the sensors is relatively high.

What happens when most of the sensors are faulty is illustrated in Figures 4.5, 4.6 and 4.7. Table 4.4 contains the results of the simulation with 1000 sensors. Notice that the final interval computed by the algorithm does not include the correct value due to the very large fraction of wildly faulty sensors. It is relevant to mention one *caveat* here. The actual form of the final solution is also dependent upon the seed of the random number generator used to start the *Monte Carlo* simulation. Thus,

Table 4.4: 1000 sensors; 800 wildly faulty, 100 tamely faulty

Sampling Frequency	Resolution Level	Sensor Interval		
		Start Point	End Point	Width
coarse	3	0.252377	1.006563	0.754186
coarse	4	0.252377	1.006563	0.754186
medium	5	-0.047790	0.203101	0.250891
medium	6	-0.047790	0.203101	0.250891
medium	7	-0.016398	0.021779	0.038177
fine	8	-0.012628	-0.006013	0.006615
fine	9	-0.012628	-0.006013	0.006615

it is not unusual for the final solution of one particular simulation experiment, with almost all of the sensors being faulty, to contain the correct value.

Figures 4.8, 4.9 and 4.10 shows the sampled overlap function, at various stages of resolution, corresponding to the data in Table 4.4. Figure 4.9 shows that the region of the overlap function sampled does not change throughout the medium frequency range as also evidenced by the three rows in table 4.4 marked *medium* under the sampling frequency column.

4.5 Concluding Remarks

The concept of multiresolution has been applied to one-dimensional sensor estimates to obtain a fault-tolerant integrated estimate of the parameter being measured by the sensors. This is done by computing the overlap function of the sensors, and resolving this function at increasingly finer dyadic scales to obtain a sequence of functions, each of which consists of a series of peaks. In each of these functions, the highest peak with the largest spread is chosen and further resolved at a finer scale to obtain the next function. This process is repeated finitely many

times up to a certain scale and the sensor estimate is taken to be the region over which the maximum value of the largest peak is attained.

This method helps isolate the neighborhood of the correct value of the parameter being measured, by taking advantage of the fact that the maximum intersection of intervals occurs about the correct value, and this corresponds to the highest peaks, and further, the tamely faulty sensors cluster around the correct value, contributing to the height and spread of the peak under which the correct value lies. At each resolution, only the relevant details of $O(x)$ are considered, increasing the efficiency of the computation.

The underlying idea of this method, namely the recognition and isolation of the most prominent and robust peaks in a region and the consequent elimination of narrower and less prominent peaks as 'errors,' can indeed be used elsewhere to isolate the important characteristics of a signal and remove the 'noise' in a computationally efficient manner. One such application is in image processing, for the efficient segmentation of images with multimodal histograms. This method can be generalized, with some modifications, to multidimensional sensors and signals.

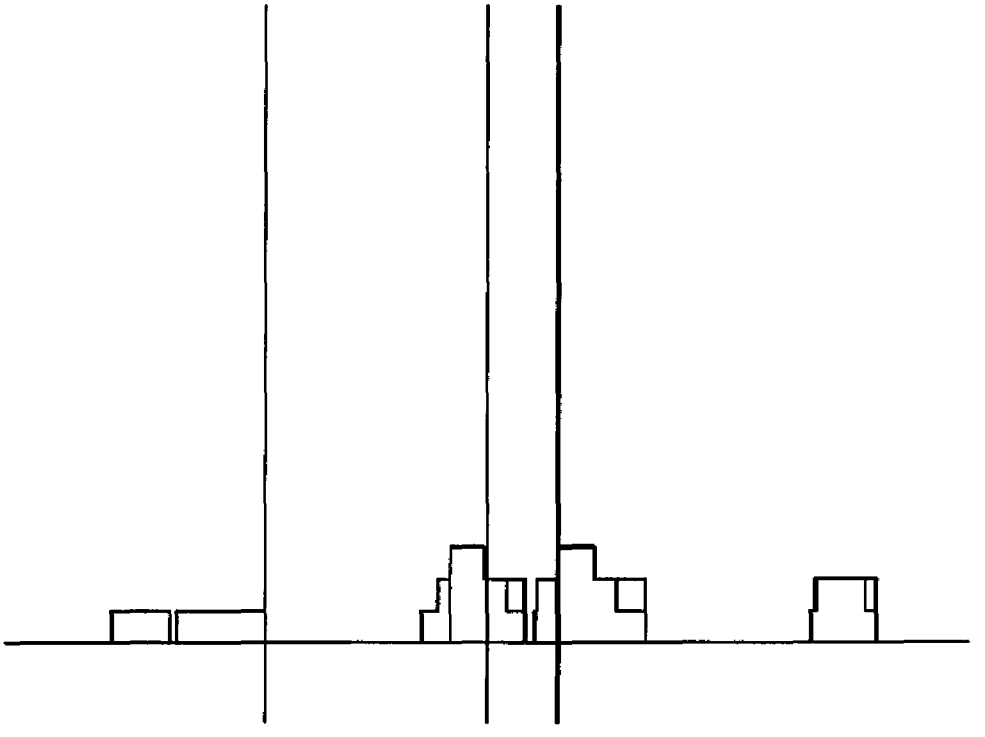


Figure 4.1: 10 sensors, 5 wild faults, 3 tame faults

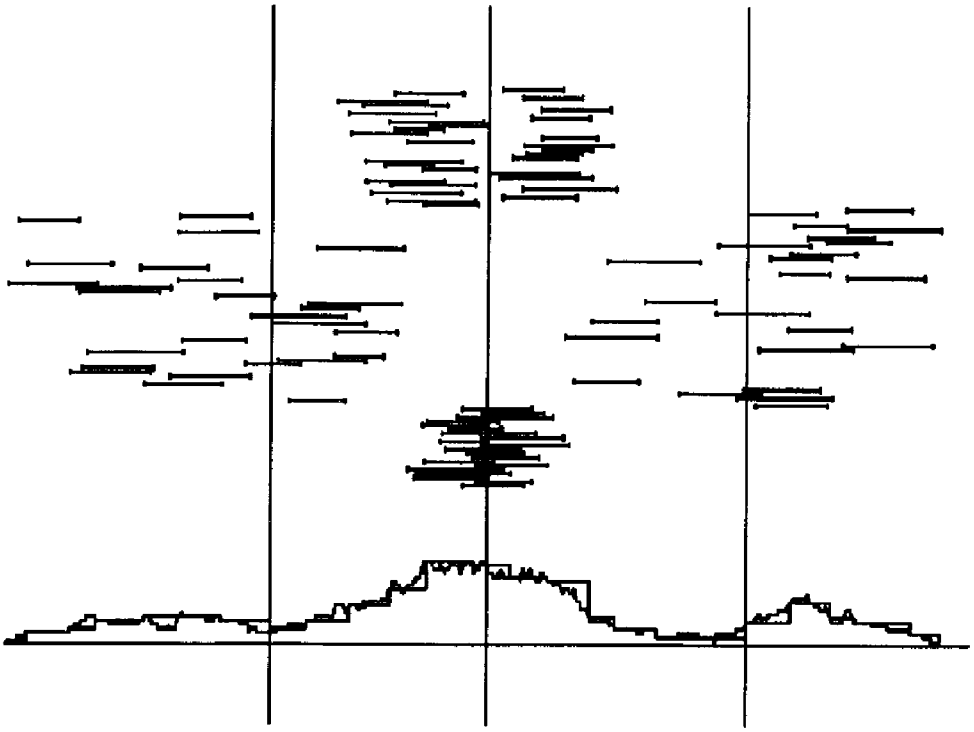


Figure 4.2: 100 sensors, 50 wild faults, 30 tame faults

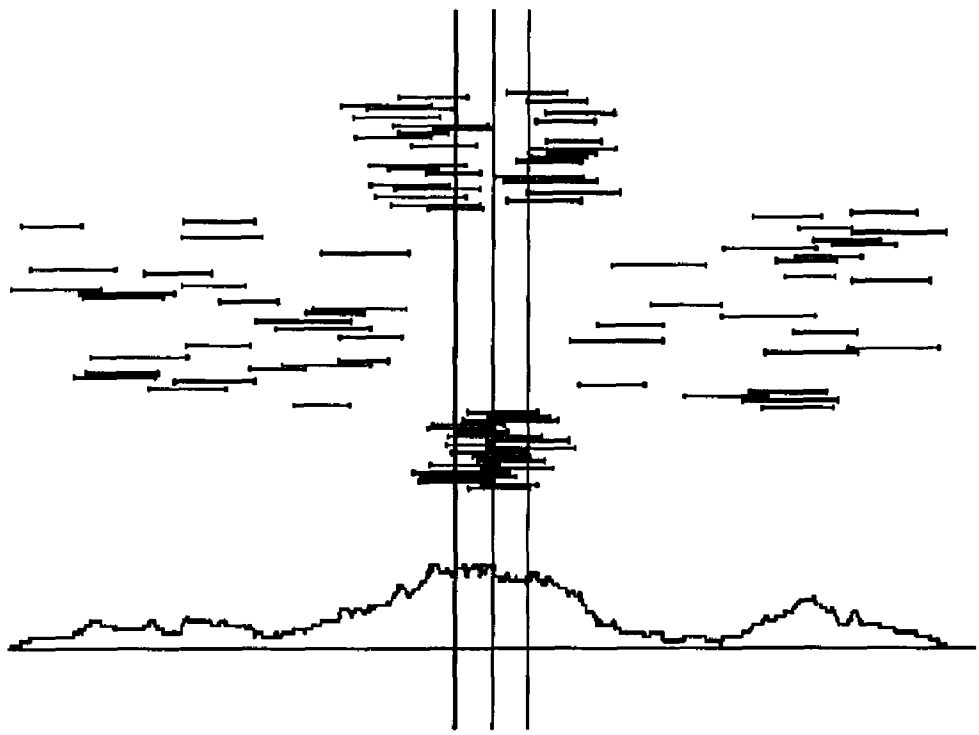


Figure 4.3: 100 sensors, 50 wild faults, 30 tame faults

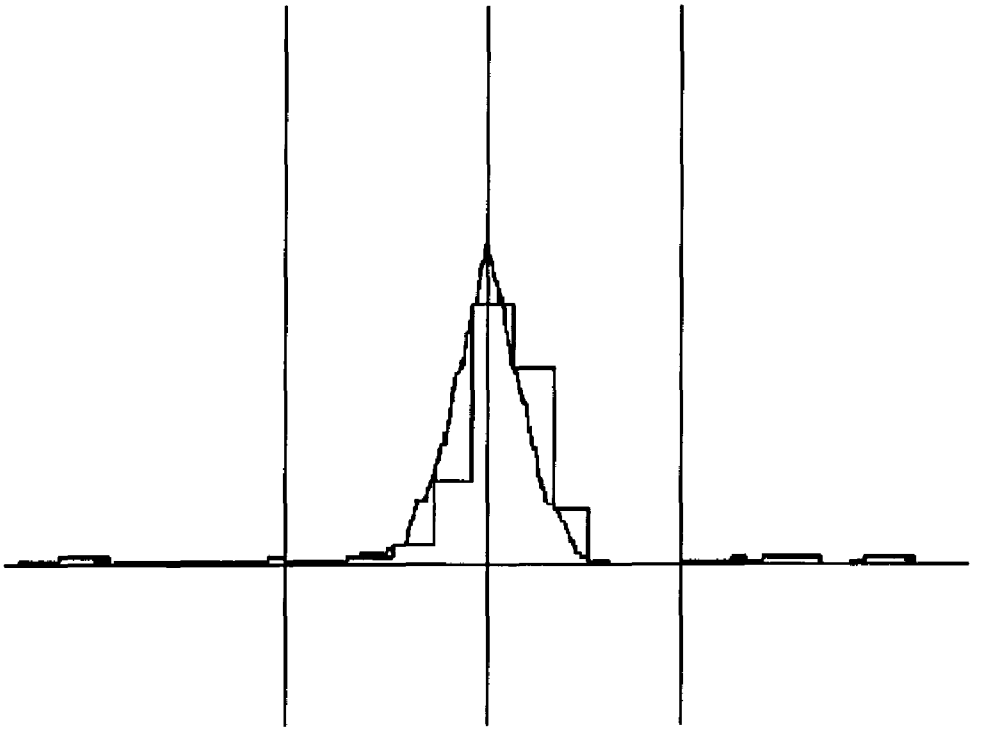


Figure 4.4: 100 sensors, 10 wild faults, 10 tame faults

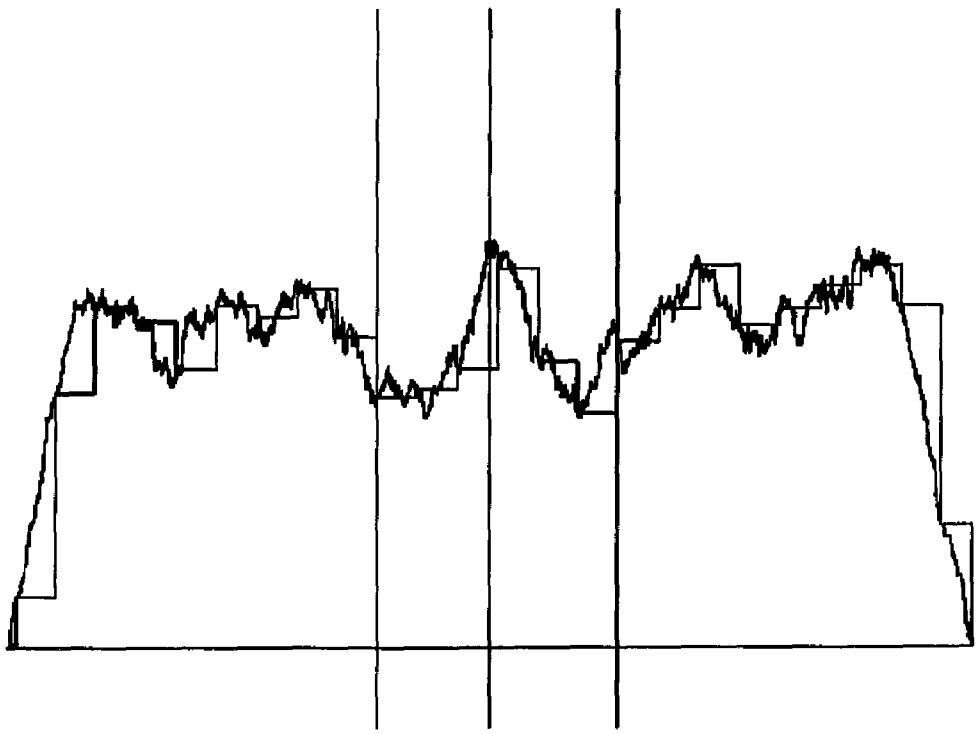


Figure 4.5: 1000 sensors, 800 wild faults, 100 tame faults

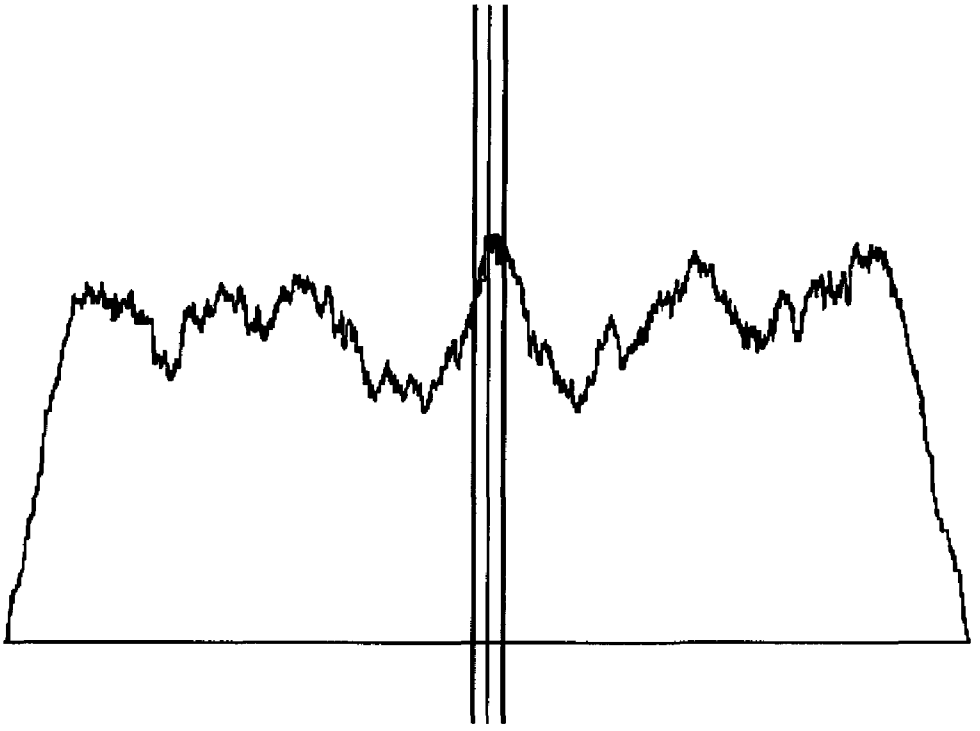


Figure 4.6: 1000 sensors, 800 wild faults, 100 tame faults

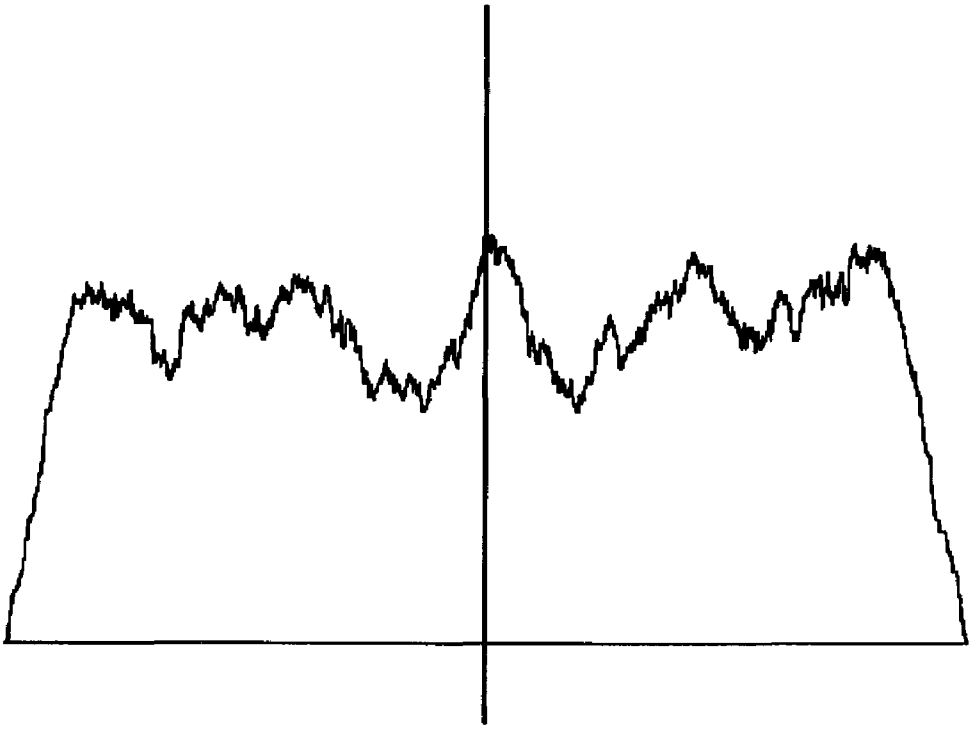


Figure 4.7: 1000 sensors, 800 wild faults, 100 tame faults

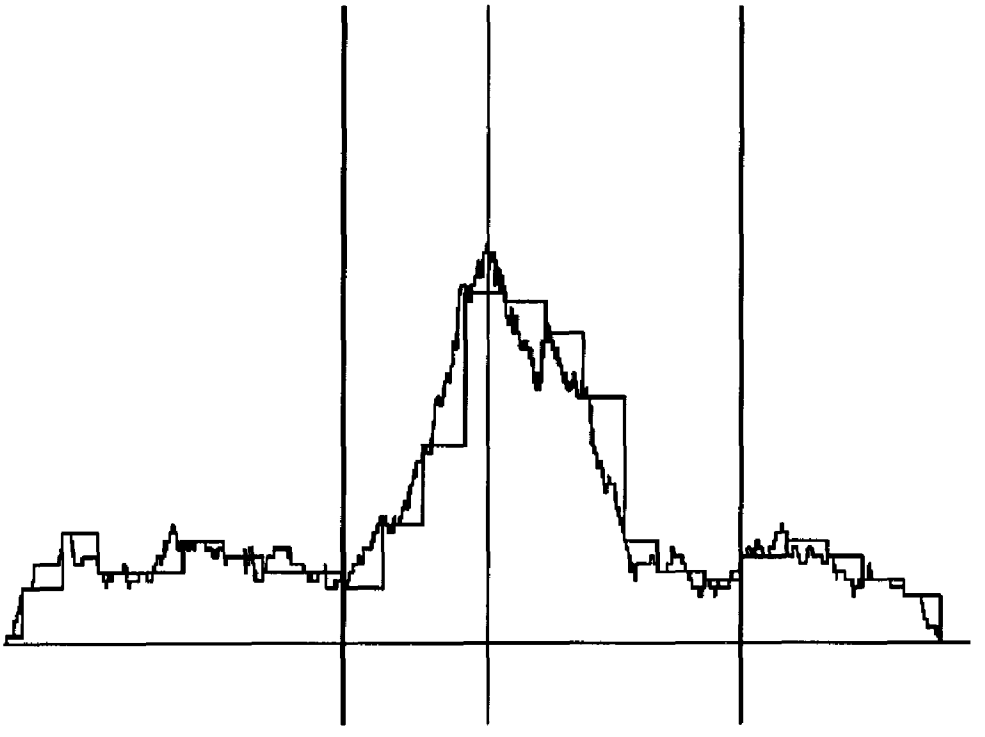


Figure 4.8: 200 sensors, 100 wild faults, 50 tame faults

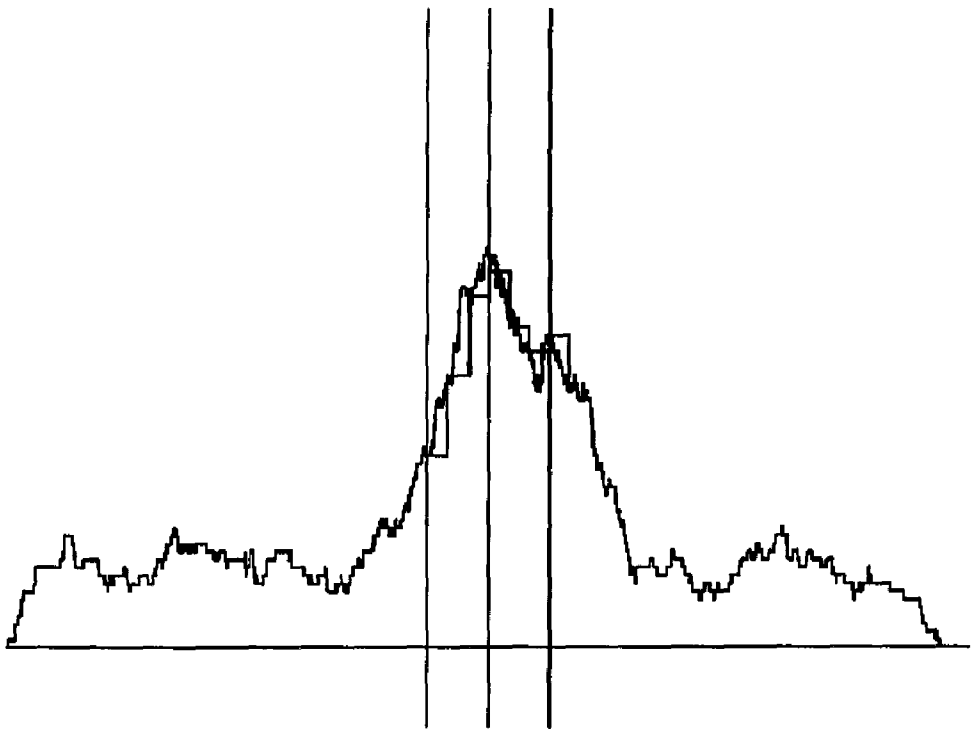


Figure 4.9: 200 sensors, 100 wild faults, 50 tame faults

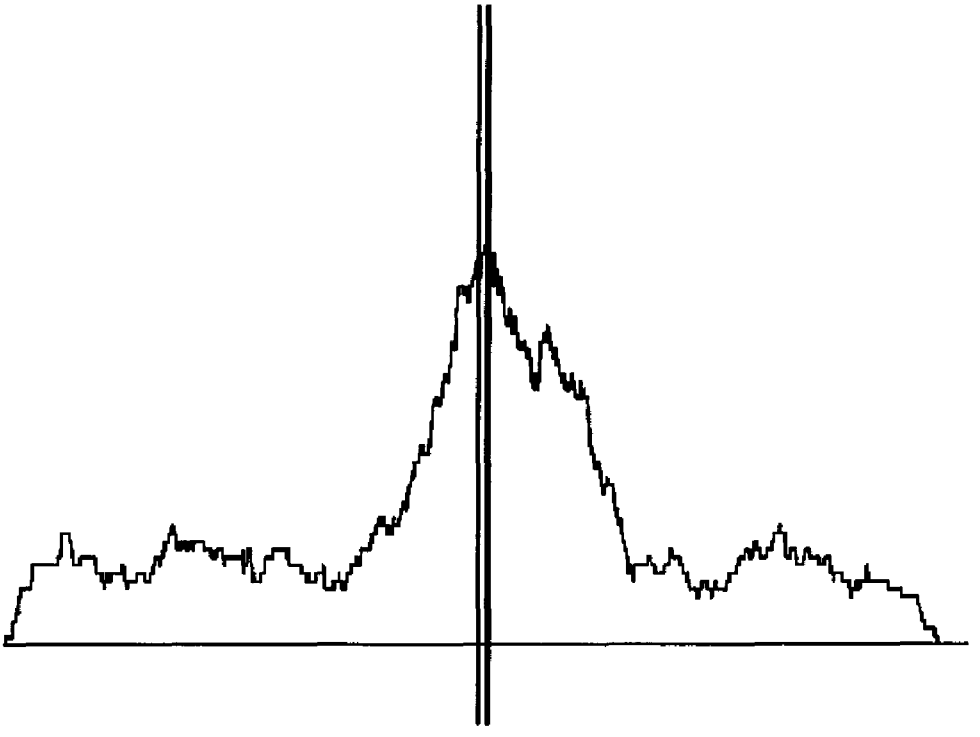


Figure 4.10: 200 sensors, 100 wild faults, 50 tame faults

Chapter 5

Multiresolutional Object Recognition in Images

5.1 Introduction

Image processing can be broadly divided into three classes:

1. Low-level processing
2. Intermediate-level processing
3. High-level processing

Low-level processing involves pixel-level image processing such as noise reduction, histogram equalization, deblurring, contrast adjustment etc., to enhance or modify the quality or appearance of images by manipulating the intensities of the image pixels.

Intermediate-level processing involves the extraction of global image artifacts such as connected components or the segmentation of an image into a few chosen gray levels. Morphological filtering of images and extraction of boundaries of connected regions also fall into this category.

High-level processing involves image understanding, in the sense of the recognition and interpretation of objects and features in images. This subsumes the searching of an image for macrostructures that act as cues for the identification and location of various image features, and labeling them as identified. While techniques for most low and intermediate-level operations on images are well understood

and even perfected, the realm of high-level image processing is still an imprecisely understood and a largely speculative area. There is a lot of uncertainty in our knowledge of the Human cognitive processes that enable us to understand complex images with such ease. As a result, most models of image understanding are highly restrictive and simplified, making their range of applicability very narrow.

Another issue that plagues high-level image processing is the computational burden involved. The most important applications of high-level image processing are in robotic vision, industrial automation and military applications such as remote surveillance and reconnaissance. These applications require real-time high-level image processing capability for quick reaction and effective performance. This makes the computational efficiency of a working model of image understanding gain priority over its generality or range of applicability, as a measure of its usefulness.

In this part of the dissertation, the problem of object and shape recognition in images is addressed. This problem has been posed in several forms, and has been attempted from several points of view (see [31, 32, 35, 36, 37, 44, 45, 46, 47, 52, 62, 61, 64]). By far the most common and successful approach to object recognition in an image, is by *template matching.* Template matching is a basic operation of high-level image processing, used to detect image features that are sought in an image. A *template* is a representative "picture" of a feature or object of interest. It is matched with different parts of an image, and the degree of match is quantitatively measured to see if the object in the template is present in the image. The template used usually contains an "expected" object or feature that gives desired information regarding the image and its contents. That is, the template belongs to a bank of finitely many templates, each of which represents an object/feature of interest for a

specific application. In general the image is assumed to contain features or objects of a certain class, represented by the template set used in the application.

Template matching for recognition problems in images, albeit popular, is seldom computationally efficient for serious implementation of any consequence. As objects in images can occur at any scale and orientation with respect to their template representations, a trial and error search in scale-orientation space has to be performed by transforming each test template by various scales and rotations, and for each such transformation, matching the template at each pixel in the image for a fit. Clearly this can lead to an explosive search, consuming enormous amount of computational time. Therefore, most often the template set is restricted to contain templates of a few rudimentary image features such as corners, edges and forks, that are relatively invariant to scale variations.

A significant step towards the simplification of template matching was proposed by Wong and Hall [52]. They put forward a hierarchical matching technique that greatly reduced the number of sites in the image at which a given template must be matched to check for the presence or absence of the object contained in the template. Their method however does not address the scale and orientation variations between an object in an image and its template representation.

Anisimov and Gorsky [63] proposed a search technique that is logarithmically efficient in the search for orientation, but does not account for scale variations.

In all the methods proposed, two common components that are responsible for computational intensity of the methods are i) the search for the proper scale and rotation of each test template with respect to its matching object in the image (which may not even be present, in which case the search is wasteful,) and ii) the number of templates that must be tested for a match in the image.

The method proposed in what follows, attempts to simplify the process of template matching by reducing the choice of templates for a given image, and by eliminating the search in *pose* (scale-orientation-translation) space, altogether. This is done by preprocessing a given image to extract geometric information regarding the objects it contains. This information is used to guide the choice of appropriate templates, transform them to the correct scale and orientation, and locate the position in the image at which the match is to be performed for a proper fit. The object recognition paradigm adopted here assumes that i) all templates used for matching belong to a template data base representing objects that are of interest to the application, and that ii) all the images considered contain only those objects that are represented in the template data base.

In order to reduce the computations, the preprocessing of an image is done on a smaller and lower resolution approximation of the original image, by decomposing the image in a wavelet-based image pyramid. A wavelet-based pyramidal decomposition of an image has several desirable properties that make it superior to other pyramidal decompositions:

1. A wavelet-based image pyramid is spatially optimal, i.e., the pyramidal decomposition takes the same amount of space as the original image.

2. The original image is exactly recoverable from its pyramid.

3. A fast algorithm that approximates the image at any scale, exists.

4. The pyramid contains edge information of the image at various scales, which can be used to obtain cues to identify the geometric features of the objects in an image.

Wavelet analysis is a very recent area with rapidly growing applications. In particular, it has gained tremendous appeal in signal and image processing applications in a very short time, and this trend is on a steady rise. We briefly introduce wavelet analysis in a semi-rigorous setting, followed by a description of wavelet transforms and pyramidal decomposition of images. We then address the problem of multiresolutional template matching with the tacit assumption that the matching is done at several resolutions on the image using the pyramidal decompositions of both the image and the templates.

Chapter 6

Wavelet Analysis

6.1 Introduction

Wavelet analysis is an area of mathematical analysis that has had origins in certain areas of Mathematics, Physics and Engineering, where it was independently anticipated in various forms. However, as a coherent and consolidated branch of analysis, it is of very recent origin. Its youth notwithstanding, it has rapidly become an extensively applied tool in several fields. In particular, signal and image processing have gained tremendous impetus from wavelet-based techniques, and in a large number of applications, wavelet analysis has consistently and convincingly outdone Fourier analysis.

We briefly develop only the most relevant aspects of wavelet analysis in order to give a modest grasp of the principles of the subject, and also to prepare ground for the derivation of the pyramidal decomposition of images with respect to an orthonormal wavelet basis. We start with a few basic definitions.

6.2 Preliminaries

A window function $w(t)$ is a function in $L^2(\mathbf{R})$ such that both w and $\hat{w}$ have rapid decay (i.e., w is well localized in time, while $\hat{w}$ is well localized in frequency). Multiplying a signal by a window function before taking its Fourier transform has the effect of restricting the spectral information of the signal to the domain of

influence of the window function. Using translates of the window function on the time axis to cover the entire time domain, a signal can be analyzed for spectral information in localized neighborhoods in time.

Definition : *A nontrivial function* $w \in L^2(\mathbf{R})$ *is called a window function if* $tw(t) \in L^2(\mathbf{R})$.

Remark : $tw(t) \in L^2(\mathbf{R}) \Rightarrow \mid t \mid^{1/2} w(t) \in L^2(\mathbf{R})$.

Therefore, writing

$$w(t) = (1+ \mid t \mid)^{-1}(1+ \mid t \mid)w(t)$$

and applying Schwartz inequality, we have

$$\| w \|_1 \leq \| (1+ \mid t \mid)^{-1} \|_2 \| (1+ \mid t \mid) \|_2 < \infty$$

Hence $w \in L^1(\mathbf{R})$. This implies that $\hat{w}$ is continuous. By Parseval's identity, $\hat{w} \in L^2(\mathbf{R})$. However, in general it is not true that $\omega\hat{w}(\omega) \in L^2(\mathbf{R})$. In other words, it is possible that while w is a window function, $\hat{w}$ is not. An example of such a window function is the Haar function $\psi_{\mathbf{H}}$:

$$\psi_{\mathbf{H}}(\mathbf{x}) = \begin{cases} 1 & \text{if } 0 \leq x < \frac{1}{2} \\ -1 & \text{if } \frac{1}{2} \leq x < 1 \\ 0 & \text{otherwise} \end{cases}$$

Definition : *The* **center** t^* *and* **radius** Δ_w *of a window function* w *are defined by*

$$t^* = \| w \|_2^{-2} \int_{-\infty}^{\infty} t \mid w(t) \mid^2 dt,$$

and

$$\Delta_w = \| w \|_2^{-1} \left[\int_{-\infty}^{\infty} (t - t^*)^2 \mid w(t) \mid^2 dt \right]^{1/2}$$

respectively. $2\Delta_w$ *is the width of the window function* w.

6.3 The Integral Wavelet Transform

Define an integral transform on $L^2(\mathbf{R})$ by:

$$(\mathcal{W}_\psi f)(a,b) = \int_{-\infty}^{\infty} f(t)\overline{\psi_{a,b}(t)}dt$$

where $\psi_{a,b}(t) = \mid a \mid^{-1/2} \psi(\frac{t-b}{a})$, with $a, b \in \mathbf{R}$, $a \neq 0$ and $\psi \in L^2(\mathbf{R})$, Further ψ satisfies the condition

$$\int_{-\infty}^{\infty} \psi(t)dt = 0$$

ψ is called the *(mother) wavelet* and the $\psi_{a,b}$ are called *wavelets.* $\mathcal{W}_\psi$ is called the *integral wavelet transform* with respect to the *wavelet* ψ.

If both the *wavelet* ψ and its Fourier transform $\hat{\psi}$ are window functions with centers t^* and ω^*, and radii Δ_ψ and $\Delta_{\hat{\psi}}$ respectively, then the integral wavelet transform

$$(\mathcal{W}_\psi f)(a,b) = \mid a \mid^{-1/2} \int_{-\infty}^{\infty} f(t)\overline{\psi_{a,b}(t)}dt$$

of a signal $f \in L^2(\mathbf{R})$ localizes it within a time window given by $[at^* + b - a\Delta_\psi, at^* + b + a\Delta_\psi]$. If $\vartheta(\omega) = \hat{\psi}(\omega + \omega^*)$, then clearly, ϑ is also a window function, with center 0 and radius $\Delta_{\hat{\psi}}$. Moreover

$$(\mathcal{W}_\psi f)(a,b) = \frac{a \mid a \mid^{-1/2}}{2\pi} \int_{-\infty}^{\infty} \hat{f}(\omega)\overline{\vartheta(\omega - a^{-1}\omega^*)}e^{ib\omega}d\omega$$

(by Parseval's identity). This means that $\mathcal{W}_\psi$ localizes the spectrum of f to a *frequency window* given by $[a^{-1}\omega^* - a^{-1}\Delta_{\hat{\psi}}, a^{-1}\omega^* + a^{-1}\Delta_{\hat{\psi}}]$. Thus we have a *time-frequency* window for analyzing finite energy analog signals with the help of the integral wavelet transform $\mathcal{W}_\psi$. Since only positive frequencies are of interest, the wavelet function ψ is chosen with $\omega^* > 0$.

The width (time window size) $2a\Delta_\psi$ of the time-frequency window is inversely proportional to the center frequency $a^{-1}\omega^*$, while the height (frequency window

size) $2a^{-1}\Delta_{\hat{\psi}}$ is directly proportional to the center frequency. Thus the area of the time-frequency windows is constant, and is given by $4\Delta_{\psi}\Delta_{\hat{\psi}}$, The ratio of the center frequency to the width of the frequency band is also a constant, given by

$$\frac{a^{-1}\omega^*}{2a^{-1}\Delta_{\hat{\psi}}} = \frac{\omega^*}{2\Delta_{\hat{\psi}}}$$

Henceforth we assume that both ψ and $\hat{\psi}$ are window functions. We also use the variables t and x, and ω and ξ interchangeably. We interpret t and ω as the time and frequency variables, respectively, and x and ξ as the space and scale variables, respectively. Thus f is appropriately a time-varying signal, or a function defined over space. For the wavelet transform to be invertible, the basic wavelet ψ must satisfy the *admissibility criterion*

$$\int_{-\infty}^{\infty} \frac{|\hat{\psi}(\xi)|^2}{|\xi|} d\xi < \infty$$

Theorem : *If $f, g \in L^2(\mathbf{R})$, then the following resolution of the identity formula holds:*

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} (\mathcal{W}_{\psi}f)(a,b)\overline{(\mathcal{W}_{\psi}g)(a,b)}a^{-2}dadb = C_{\psi}\langle f, g\rangle$$

where

$$C_{\psi} = \int_{-\infty}^{\infty} \frac{|\hat{\psi}(\xi)|^2}{|\xi|} d\xi$$

The following corollary gives the *inverse wavelet transform*

Corollary 1:

$$f = C_{\psi}^{-1}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} (\mathcal{W}_{\psi}f)(a,b)\psi_{a,b}a^{-2}dadb$$

where the equality is meant in the sense that taking the inner products on both sides of the above equation with any function in $L^2(\mathbf{R})$ yields an identity.

Indeed, the equality in the above corollary is valid in a stronger sense:

Corollary 2:

$$\lim_{\substack{\alpha\to 0\\ \beta,\gamma\to\infty}} \| f - C_\psi^{-1} \int_{-\gamma}^{\gamma}\int_{-\alpha}^{\beta} (\mathcal{W}_\psi f)(a,b)\psi_{a,b}a^{-2}dadb \|_2 = 0$$

Remark : If ψ is an admissible wavelet, then

$$\int_{-\infty}^{\infty} \psi(x)dx = 0$$

Indeed, since ψ is a window function, $\hat{\psi}$ is continuous. Since ψ is admissible, $C_\psi < \infty$. Therefore $\hat{\psi}(0) = 0$ *i.e.*,

$$\int_{-\infty}^{\infty} \psi(x)dx = 0$$

6.4 The Discrete Wavelet Transform

The continuous (integral) wavelet transform involved wavelets of the form

$$\psi_{a,b}(x) = \mid a \mid^{-1/2} \psi(\frac{x-b}{a})$$

with $a, b \in \mathbf{R}, a \neq 0$ and ψ is admissible ($\int_{-\infty}^{\infty} \frac{|\hat{\psi}(\zeta)|^2}{\zeta} d\zeta < \infty$).

As we have already seen, the integral wavelet transform, (unlike the Fourier transform,) has very good time-frequency localization properties. Moreover, the functions $\psi_{a,b}$ belong to $L^2(\mathbf{R})$.

It is possible to extract a discrete subset of the set of functions

$$\{\psi_{a,b}\}_{(a,b)\in\mathbf{R}^*\times\mathbf{R}'}$$

which forms a basis of $L^2(\mathbf{R})$ and inherits the time frequency localization property of the continuous family. This involves the singling out of an appropriate lattice of the set $\mathbf{R}^* \times \mathbf{R}(\mathbf{R}^* = \mathbf{R}\backslash\{0\})$ and selecting wavelets parametrized by elements of this lattice.

In signal processing applications, one is interested in nonnegative frequencies only. Since the concept of frequency corresponds to that of feature scale in image processing, we would like to consider only positive values of the frequency (scale) parameter a . With this restriction, the admissibility criterion becomes

$$\int_0^\infty \frac{|\hat{\psi}(\zeta)|^2}{|\zeta|} d\zeta = \int_{-\infty}^0 \frac{|\hat{\psi}(\zeta)|^2}{|\zeta|} d\zeta < \infty$$

We have seen that the wavelet $\psi_{a,b}$ is centered at $at^* + b$ in time and has its time window width equal to $2a\Delta_\psi$, and $\hat{\psi}_{a,b}$ is centered at $a^{-1}\omega^*$ in frequency and has its frequency window width equal to $2a^{-1}\Delta_{\hat{\psi}}$. We let frequency ω be equal to $a^{-1}\omega^*$. Since $a > 0$ and $\omega^* > 0$, $\omega \in (0, \infty)$.

In order to discretize the continuous family of wavelets $\{\psi_{a,b}\}$, we first partition the frequency domain $(0, \infty)$ into dyadic intervals as follows:

$$(0, \infty) = \bigcup_{j=-\infty}^{\infty} (2^{-j}\Delta_{\hat{\psi}}, 2^{-j+1}\Delta_{\hat{\psi}})$$

Since shifting the phase of ψ by θ is equivalent to translating $\hat{\psi}$ by θ on the frequency axis, *i.e.*, $\psi'(t) = e^{i\theta t}\psi(t) \Rightarrow \hat{\psi}'(\omega) = \hat{\psi}(\omega - \theta)$, while leaving the time and frequency radii, without loss of generality, we can assume that $\omega^* = 3\Delta_{\hat{\psi}}$. This yields

$$(a^{-1}\omega^* - a^{-1}\Delta_{\hat{\psi}}, a^{-1}\omega^* + a^{-1}\Delta_{\hat{\psi}}] = (2a^{-1}\Delta_{\hat{\psi}}, 4a^{-1}\Delta_{\hat{\psi}})$$

If we discretize the dilation parameter a by letting it take the values $2^j, \forall j \in \mathbf{Z}$ then we have

$$(a^{-1}\omega^* - a^{-1}\Delta_{\hat{\psi}}, a^{-1}\omega^* + a^{-1}\Delta_{\hat{\psi}}] = (2^{-j+1}\Delta_{\hat{\psi}}, 2^{-j+2}\Delta_{\hat{\psi}}), j \in \mathbf{Z}.$$

Thus, by setting $\omega^* = 3\Delta_{\hat{\psi}}$, and discretizing the dilation parameter a as above, the frequency domains of the resulting set of wavelets form the dyadic partition of the frequency domain we constructed.

We now discretize the translation parameter b by observing that at each scale 2^j the width of the time window is $2^{j+1}\Delta_\psi$. Hence we may discretize the time domain at each scale separately by setting the distance between two consecutive wavelets' centers equal to $2^{j+1}\Delta_\psi$. This will ensure that the discrete set of wavelets will cover the entire time domain at each scale. Denoting the width $2\Delta_\psi$ of the time window of ψ by b_0, we have b taking the discrete range of values $k2^j b_0, k \in \mathbf{Z}$.

Introducing the notation $\psi_{j,k}(t) = 2^{-j/2}\psi(2^{-j}t - kb_0),\ \forall j,k \in \mathbf{Z}$, we have a discrete set of wavelets whose time-frequency windows cover the whole of the ω-t plane. The choice of b_0 is arbitrary, and it can be replaced by any other real number by appropriately rescaling the basic wavelet ψ. We therefore set $b_0 = 1$.

The wavelet transform $\mathcal{W}_\psi$ of any function $f \in L^2(\mathbf{R})$, restricted to this discrete set of wavelets, is given by

$$(\mathcal{W}_\psi f)(\frac{1}{2^j}, \frac{k}{2^j}) = 2^{-j/2}\int_{-\infty}^{\infty} f(x)\overline{\psi_{j,k}(x)}dx = \langle f, \psi_{j,k}\rangle\ \forall j,k \in \mathbf{Z}$$

Thus the *discrete wavelet coefficients* of any $f \in L^2(\mathbf{R})$ are given by

$$\langle f, \psi_{j,k}\rangle, j,k \in \mathbf{Z}.$$

The motivation behind the discretization of the continuous wavelet transform is two-fold: i) Not only is any function $f \in L^2(\mathbf{R})$ completely characterized by its discrete wavelet coefficients $\langle f, \psi_{j,k}\rangle, j,k \in \mathbf{Z}$, but ii) it is also possible to recover any function in $L^2(\mathbf{R})$ from its discrete wavelet coefficients in a numerically stable way.

In other words, $L^2(\mathbf{R})$ is "spanned" by the discrete set of wavelets $\{\{\psi_{j,k}\}_{j,k\in\mathbf{Z}}\}$ for appropriate choice of ψ and b_0, *and* there exists an algorithm to determine the discrete wavelet coefficients of a function $f \in L^2(\mathbf{R})$.

The discrete set of wavelets $\{\psi_{j,k}\}_{j,k\in\mathbf{Z}}$ characterize a function $f \in L^2(\mathbf{R})$ by means of the discrete wavelet coefficients $\langle f, \psi_{j,k}\rangle \quad j,k \in \mathbf{Z}$, if

$$\langle f, \psi_{j,k}\rangle = \langle g, \psi_{j,k}\rangle \forall j,k \in \mathbf{Z} \Leftrightarrow f = g,$$

or equivalently,

$$\langle f, \psi_{j,k}\rangle = 0 \; \forall j,k \in \mathbf{Z} \Leftrightarrow f = 0.$$

The characterization is *numerically stable*, in the sense that small perturbations in the wavelet coefficients $\langle f, \psi_{j,k}\rangle$ of f correspond to small perturbations of the function f in the L^2-norm.

For any basic wavelet ψ that is admissible, and has good decay in time and frequency, and satisfies $\hat{\psi}(0) = 0$,

$$\sum_{j,k\in\mathbf{Z}} | \langle f, \psi_{j,k}\rangle |^2 \leq C \; || f ||_2^2$$

for some constant C. This implies that the sequence

$$\{\langle f, \psi_{j,k}\rangle\}_{j,k\in\mathbf{Z}}$$

of discrete wavelet coefficients of $f \in L^2(\mathbf{R})$ w.r.t. the discretized family of wavelets $\{\psi_{j,k}\}_{j,k\in\mathbf{Z}}$ belongs to

$$l^2(\mathbf{Z}^2) = \left\{ \{c_{i,j}\}_{i,j\in\mathbf{Z}} : \sum_{i,j\in\mathbf{Z}} | c_{i,j} |^2 < \infty \right\}$$

Thus $f \mapsto \{\langle f, \psi_{j,k}\rangle\}_{j,k\in\mathbf{Z}}$ is a mapping from $L^2(\mathbf{R})$ into $l^2(\mathbf{Z}^2)$.

Now $L^2(\mathbf{R})$ is equipped with the metric topology induced by the norm $|| \; ||_2$ on $L^2(\mathbf{R})$, while $l^2(\mathbf{Z}^2)$ has the metric topology induced by the norm

$$|| \, c \, ||^2 = \sum_{i,j\in\mathbf{Z}} | \, c_{i,j} \, |^2 \; \forall c \in l^2(\mathbf{Z}^2)$$

Therefore we can take the numerical stability of the characterization of functions in $L^2(\mathbf{R})$ by their sequences of discrete wavelet coefficients to mean: whenever the discrete wavelet coefficient sequences of two functions are close in $l^2(\mathbf{Z}^2)$, the functions themselves are close in $L^2(\mathbf{R})$. That is, if

$$\sum_{i,j\in\mathbf{Z}} |\langle f,\psi_{i,k}\rangle|^2$$

is small, then $\|f\|_2^2$ is small.

In particular, $\exists\alpha<\infty \ni \sum_{i,j\in\mathbf{Z}} |\langle f,\psi_{i,j}\rangle|^2 \leq 1 \Rightarrow \|f\|^2 \leq \alpha$

For $f\in L^2(\mathbf{R})$, define

$$\tilde{f} = f\left[\sum_{i,j\in\mathbf{Z}} |\langle f,\psi_{i,j}\rangle|^2\right]^{-1/2}$$

Then

$$\sum_{i,j\in\mathbf{Z}} |\langle \tilde{f},\psi_{i,k}\rangle|^2 \leq 1 \text{ and } \|f\|^2 < \alpha$$

$$\therefore \sum_{i,j\in\mathbf{Z}} |\langle f,\psi_{i,k}\rangle|^2 \geq \alpha^{-1}\|f\|^2 .$$

i.e.,

$$A\|f\|^2 \leq \sum_{i,j\in\mathbf{Z}} |\langle f,\psi_{i,j}\rangle|^2,$$

for $A>0$. This is the condition for numerical stability of the characterization of $f\in L^2(\mathbf{R})$ by its discrete wavelet coefficients.

For a numerically stable characterization of the functions in $L^2(\mathbf{R})$ by the discrete set of wavelets $\{\psi_{j,k}\}_{j,k\in\mathbf{Z}}$ it is necessary and sufficient that there exist constants $0<A\leq B<\infty$ such that

$$A\|f\|^2 \leq \sum_{i,j\in\mathbf{Z}} |\langle f,\psi_{i,j}\rangle|^2 \leq B\|f\|^2, \quad \forall f\in L^2(\mathbf{R})$$

For appropriate choices of the wavelet ψ, the set of wavelets

$$\left\{\psi_{j,k}(x) = 2^{-\frac{j}{2}}\psi\left(2^{-j}x-k\right), k,j\in\mathbf{Z}\right\}$$

form an orthonormal basis of $L^2(\mathbf{R})$. If we restrict the above set to a fixed value j of the scale parameter, then denoting by $\mathbf{W}_j$ the closure of the span of the resulting subbasis $\{\psi_{j,k}\}_{k\in\mathbf{Z}}$, *i.e.*,

$$Closure_{L^2(\mathbf{R})}\left[Span_{\mathbf{C}}\left(\{\psi_{j,k}\}\right)\right] = \mathbf{W}_j,$$

we have the direct sum decomposition

$$L^2(\mathbf{R}) = \bigoplus_{j\in\mathbf{Z}} \mathbf{W}_j$$

If we let

$$\mathbf{V}_J = \bigoplus_{j=J+1}^{\infty} \mathbf{W}_j, \forall J \in \mathbf{Z}$$

Then the subspaces $\{\mathbf{V}_j\}_{j\in bfZ}$ form a nested sequence of subspaces of $L^2(\mathbf{R})$, and impose a geometric structure on it, called a *multiresolution analysis.*

Multiresolution analyses on $L^2(\mathbf{R})$are a very elegant and natural framework for understanding and constructing wavelet bases, first formulated by S. Mallat and Y. Meyer.

6.5 Multiresolution Analysis (MRA) of $L^2(\mathbf{R})$

Definition : *An MRA of* $L^2(\mathbf{R})$ *is a sequence of closed subspaces* $\{\mathbf{V}_j\}_{j\in\mathbf{Z}}$ *of* $L^2(\mathbf{R})$, *satisfying the following properties :*

- *i)* $\mathbf{V}_j \subset \mathbf{V}_{j-1}\ \forall j \in \mathbf{Z}$; *(nesting property)*
- *ii)* $Closure_{L^2(\mathbf{R})}\left(\bigcup_{j\in\mathbf{Z}} \mathbf{V}_j\right) = L^2(\mathbf{R})$; *(density of the union in* $L^2(\mathbf{R})$*)*
- *iii)* $\bigcap_{j\in\mathbf{Z}} \mathbf{V}_j = \{0\}$;
- *iv)* $f(x) \in \mathbf{V}_j \Leftrightarrow f(2x) \in \mathbf{V}_{j-1}\ \ \forall j \in \mathbf{Z}$; *(scaling property)*

- *v)* $f(x) \in \mathbf{V}_0 \Rightarrow f(x-n) \in \mathbf{V}_0 \; \forall n \in \mathbf{Z}$; *(invariance under integral translations)*

- *vi)* $\exists \phi \in \mathbf{V}_0 \ni \{\phi_{0,n}\}_{n \in \mathbf{Z}}$ *is an orthonormal basis of* $\mathbf{V}_0$, *where* $\phi_{j,k}(x) = 2^{-j/2}\phi(2^{-j}x - k) \;\; \forall j,k \in \mathbf{Z}$; *(existence of a scaling function).*

Condition iv) implies that each of the subspaces $\mathbf{V}_j$ is a scaled version of the central subspace $\mathbf{V}_0$, and together with v) and vi) it implies that $\{\phi_{j,k}\}_{k \in \mathbf{Z}}$ is an orthonormal basis for $\mathbf{V}_j \; \forall j \in \mathbf{Z}$.

$\forall j \in \mathbf{Z}$, denote by $\mathbf{W}_j$ the orthogonal complement of $\mathbf{V}_j$ in $\mathbf{V}_{j-1}$. Then $\mathbf{V}_{j-1} = \mathbf{V}_j \oplus \mathbf{W}_j$, where $\oplus$ is the direct sum, and $\forall u \in \mathbf{V}_{j-1}, u = u + w$, where $v \in \mathbf{V}_j$ and $w \in \mathbf{W}_j$ with $\langle v, w \rangle = 0$; also the decomposition of u is unique:

$\mathbf{W}_j \perp \mathbf{W}_{j'} \; \forall j \neq j'$, since if $j > j'$, then $\mathbf{W}_{j'} \subset \mathbf{V}_j \perp \mathbf{W}_j$ Thus if $j < i$, $\mathbf{V}_j = \mathbf{V}_i \oplus \bigoplus_{k=0}^{i-j-1} \mathbf{W}_{i-k}$.

As $Closure_{L^2(\mathbf{R})}(\bigcup_{j \in \mathbf{Z}} \mathbf{V}_j) = L^2(\mathbf{R})$ and $\bigcap_{j \in \mathbf{Z}} = \{0\}$, $L^2(\mathbf{R}) = \bigoplus_{j \in \mathbf{Z}} \mathbf{W}_j$, where the $\mathbf{W}_j$ are mutually orthogonal closed subspaces of $L^2(\mathbf{R})$.

Denote by $\mathbf{P}_j$ the orthogonal projection operator of $L^2(\mathbf{R})$ onto $\mathbf{V}_j$, and by $\mathbf{Q}_j$ the orthogonal projection operator of $L^2(\mathbf{R})$ onto $\mathbf{W}_j$. Then

$$\mathbf{Q}_j = \mathbf{P}_{j-1} - \mathbf{P}_j$$

. The subspaces $\mathbf{W}_j$ inherit the scaling property from the $\mathbf{V}_j$: $f(x) \in \mathbf{W}_j \Leftrightarrow f(2x) \in \mathbf{W}_{j-1}$.

Given an MRA of $L^2(\mathbf{R})$, one can construct a basic wavelet ψ such that $\{\psi_{j,k}\}_{k \in \mathbf{Z}}$ is an orthonormal basis for $\mathbf{W}_j \;\; \forall j \in \mathbf{Z}$, and $\{\psi_{j,k}\}_{j,k \in \mathbf{Z}}$ is an orthonormal basis of $L^2(\mathbf{R})$, where $\psi_{j,k}(x) = 2^{-j/2}\psi(2^{-j}x - k) \; \forall j,k \in \mathbf{Z}$.

Thus, if the sequence of closed subspaces $\{\mathbf{V}_j\}_{j\in\mathbf{Z}}$ of $L^2(\mathbf{R})$ is an MRA of $L^2(\mathbf{R})$, then

$$\exists\ \psi \in L^2(\mathbf{R}) \ni \left\{\psi_{j,k} : \psi_{j,k}(x) = 2^{-j/2}\psi(2^{-j}x - k)\ \forall j,k \in \mathbf{Z}\right\}$$

is an orthonormal basis of $L^2(\mathbf{R})$ such that

$$\mathbf{P}_{j-1} = \mathbf{P}_j + \sum_{k\in\mathbf{Z}} \langle \cdot, \psi_{j,k}\rangle \psi_{j,k} \quad \forall j \in \mathbf{Z}$$

If $\{\psi_{j,k}\}_{j,k\in\mathbf{Z}}$ is an orthonormal basis of $L^2(\mathbf{R})$, then since

$$\mathbf{P}_j : L^2(\mathbf{R}) \to \mathbf{V}_j$$

is surjective,

$$\mathbf{Q}_j = \mathbf{P}_{j-1} - \mathbf{P}_j$$

is also surjective and

$$\mathbf{Q}_j f = \sum_{k\in\mathbf{Z}} \langle f, \psi_{j,k}\rangle \psi_{j,k} \quad \forall f \in L^2(\mathbf{R})$$

Hence $\{\psi_{j,k}\}_{k\in\mathbf{Z}}$ is an orthonormal basis for $\mathbf{W}_j \quad \forall j \in \mathbf{Z}$.

Since the subspaces $\mathbf{W}_j$ have the scaling property $f(x) \in \mathbf{W}_j \Leftrightarrow f(2x) \in \mathbf{W}_{j-1}$, $\{\psi_{0,k}\}_{k\in\mathbf{Z}}$ is an orthonormal basis of $\mathbf{W}_0$ iff $\{\psi_{j,k}\}_{k\in\mathbf{Z}}$ is an orthonormal basis of $\mathbf{W}_j \quad \forall j \in \mathbf{Z}$.

The above observations imply that it is sufficient to find a function $\psi \in \mathbf{W}_0 \ni \{\psi_{0,k}\}_{k\in\mathbf{Z}}$ is an orthonormal basis of $\mathbf{W}_0$.

It is also possible to construct an MRA by first choosing an appropriate scaling function ϕ_0, and obtaining $\mathbf{V}_0$ by taking the linear span of integer translates of ϕ. The other subspaces $\mathbf{V}_j$ can be generated as scaled versions of $\mathbf{V}_0$.

6.6 Multiresolution Decomposition and Reconstruction of Functions in $\mathbf{L^2(R)}$

Given $\{\langle f, \phi_{j,k}\rangle\}_{k\in\mathbf{Z}}$ for some $f \in L^2(\mathbf{R})$, and some $j \in \mathbf{Z}$, one can rescale units and assume that $j = 0$; *i.e.*, , $\langle f, \phi_{j,k}\rangle \mapsto \langle f, \phi_{0,k}\rangle \ \ \forall k \in \mathbf{Z}$. We now compute the wavelet coefficients $\{\langle f, \psi_{j,k}\rangle\}_{k\in\mathbf{Z}} \ \ \forall j \geq 1$. In order to do this, we first require the *scaling relations* for the mother wavelet ψ and the scaling function ϕ : Since $\mathbf{V_0} \subset \mathbf{V_{-1}}$,

$$\psi = \sum_{n\in\mathbf{Z}} h_n \phi_{-1,n},$$

where

$$h_n = \langle \phi, \phi_{-1,n}\rangle.$$

Also, since $\mathbf{W_0} \subset \mathbf{V_{-1}}$,

$$\psi = \sum_{n\in\mathbf{Z}} g_n \phi_{-1,n},$$

where

$$g_n = \langle \psi, \phi_{-1,n}\rangle$$

The coefficients h_n and g_n are related by

$$g_n \ = \ (-1)^n h_{-n+1} \quad \forall n \in \mathbf{Z}.$$

The above two relations expressing the wavelet and the scaling function as a linear combination of contracted and translated versions of the scaling function, are called the "2-scale relations", or just "scaling relations". Therefore

$$\psi_{j,k}(x) = 2^{-j/2}\psi(2^{-j}x - k) = 2^{-j/2}\sum_{n\in\mathbf{Z}} g_n 2^{1/2}\phi(2^{-j+1}x - 2k - n)$$

$$= \sum_{n\in\mathbf{Z}} g_n \phi_{j-1,2k+n}(x) = \sum_{n\in\mathbf{Z}} g_{n-2k}\phi_{j-1,n}(x)$$

$$\therefore \langle f, \psi_{1,k} \rangle = \sum_{n \in \mathbf{Z}} \overline{g_{n-2k}} \langle f, \phi_{0,n} \rangle \quad \forall k \in \mathbf{Z}$$

In other words, the sequence

$$\{\langle f, \psi_{1,k} \rangle\}_{k \in \mathbf{Z}}$$

is obtained by convolution of the sequence

$$\{\langle f, \phi_{0,n} \rangle\}_{n \in \mathbf{Z}}$$

with the sequence

$$\bar{\mathbf{g}} = \{\bar{g}_{-n}\rangle_{n \in \mathbf{Z}}$$

and the retention of only the even-indexed elements of the resulting sequence. Also,

$$\forall j \in \mathbf{Z} \;\; \langle f, \psi_{j,k} \rangle = \sum_{n \in \mathbf{Z}} \overline{g_{n-2k}} \langle f, \phi_{j-1,n} \rangle \quad \forall k \in \mathbf{Z};$$

i.e., if the sequence

$$\{\langle f, \phi_{j-1,n} \rangle\}_{n \in \mathbf{Z}}$$

is known, then the sequence

$$\{\langle f, \psi_{j,k} \rangle\}_{k \in \mathbf{Z}}$$

is obtained by convolving the sequence

$$\{\langle f, \phi_{j-1,n} \rangle\}_{n \in \mathbf{Z}}$$

with the sequence

$$\bar{\mathbf{g}} = \{\bar{g}_{-n}\}_{n \in \mathbf{Z}}$$

and retaining only the even-indexed terms of the resulting sequence. Moreover, since

$$\phi_{j,k}(x) = 2^{-j/2} \phi(2^{-j}x - k) = \sum_{n \in \mathbf{Z}} h_{n-2k} \phi_{j-1,n}(x) \quad \forall j, k \in \mathbf{Z}$$

we have $\forall j \in \mathbf{Z}$,

$$\langle f, \phi_{j,k} \rangle = \sum_{n \in \mathbf{Z}} \overline{h_{n-2k}} \langle f, \phi_{j-1,n} \rangle \quad \forall k \in \mathbf{Z}.$$

Thus, for $j \geq 1$, the sequence

$$\{\langle f, \phi_{j,n} \rangle\}_{n \in \mathbf{Z}}$$

can be obtained from the sequence

$$\{\langle f, \phi_{j-1,n} \rangle\}_{n \in \mathbf{Z}}$$

by convolving it with the sequence

$$\bar{\mathbf{h}} = \{\bar{h}_{-n}\}_{n \in \mathbf{Z}}$$

and retaining only the even terms of the resulting sequence.

The following diagram illustrates the above procedure:

$$\begin{array}{ccccccc} \{\langle f, \phi_{0,n} \rangle\}_{n \in \mathbf{Z}} & \rightarrow & \{\langle f, \phi_{1,n} \rangle\}_{n \in \mathbf{Z}} & \cdots & \{\langle f, \phi_{j-1,n} \rangle\}_{n \in \mathbf{Z}} & \rightarrow & \{\langle f, \phi_{j,n} \rangle\}_{n \in \mathbf{Z}} \\ \downarrow & & \downarrow & & \downarrow & & \downarrow \\ \{\langle f, \psi_{1,k} \rangle\}_{k \in \mathbf{Z}} & & \{\langle f, \psi_{2,k} \rangle\}_{k \in \mathbf{Z}} & & \{\langle f, \psi_{j,k} \rangle\}_{k \in \mathbf{Z}} & & \{\langle f, \psi_{j+1,k} \rangle\}_{k \in \mathbf{Z}} \end{array}$$

Thus one can compute successively "coarser" approximations of f, along with the "difference in information" between successive levels of approximation.

If $f \in L^2(\mathbf{R})$, denote by f^j the projection $\mathbf{P}_j f$ of f onto the closed subspace $\mathbf{V}_j$ of $L^2(\mathbf{R})$, and denote by g^j the projection $\mathbf{Q}_j f$ of f onto the closed subspace $\mathbf{W}_j$ of $L^2(\mathbf{R})$. Then

$$f^{j-1} = f^j + g^j$$

since

$$\mathbf{V}_{j-1} = \mathbf{V}_j \oplus \mathbf{W}_j$$

and

$$f^j = g^j + g^{j+1} + L + g^{j+m} + f^{j+m}$$

since

$$\mathbf{V}_j = \bigoplus_{k=0}^{m} \mathbf{W}_{j+k} \oplus \mathbf{V}_{j+m}$$

If $a = \{a_n\}_{n\in\mathbf{Z}}$ is any sequence, then the sequence $\{a_{2n}\}_{n\in\mathbf{Z}}$ is denoted by $a\downarrow_2$. The mapping $a \mapsto a\downarrow_2$ is called *downsampling by factor* 2. Thus

$$(..., a_{-4}, a_{-3}, a_{-2}, a_{-1}, a_0, a_1, a_2, a_3, a_4, ...) \xrightarrow{\downarrow_2} (..., a_{-4}, a_{-2}, a_0, a_2, a_4, ...)$$

If $a = \{a_n\}_{n\in\mathbf{Z}}$ is any sequence, then the sequence $\{b_n\}_{n\in\mathbf{Z}}$ defined by

$$b_n = \begin{cases} a_{\frac{n}{2}} & \text{if } n \text{ is even} \\ 0 & \text{if } n \text{ is odd} \end{cases}$$

is denoted by $a\uparrow_2$. The mapping $a \mapsto a\uparrow_2$ is called *upsampling by factor* 2. Thus

$$(..., a_{-3}, a_{-2}, a_{-1}, a_0, a_1, a_2, a_3, ...) \xrightarrow{\uparrow_2}$$

$$\xrightarrow{\uparrow_2} (..., 0, a_{-3}, 0, a_{-2}, 0, a_{-1}, 0, a_0, 0, a_1, 0, a_2, 0, a_3, 0, ...)$$

If $a = \{a_n\}_{n\in\mathbf{Z}}$ is any sequence, denote by $\bar{a}$ the sequence $\{\bar{a}_n\}_{n\in\mathbf{Z}}$, and for any other sequence $b = \{b_n\}_{n\in\mathbf{Z}}$, define the action of the operator $a\#$ on the sequence b, denoted by $a\#b$, to be the sequence

$$\{\sum_{m\in\mathbf{Z}} \overline{a_{m-2n}} b_m\}_{n\in\mathbf{Z}}$$

i.e.,

$$(a\#b)_n = \sum_{m\in\mathbf{Z}} \overline{a_{m-2n}} b_m$$

Denote $c_k^j = \langle f, \phi_{j,k}\rangle$ and $d_k^j = \langle f, \psi_{j,k}\rangle$. Thus

$$c^j = \{c_n^j\}_{n\in\mathbf{Z}} = \{\langle f, \phi_{j,n}\rangle\}_{n\in\mathbf{Z}}$$

and

$$d^j = \{d_n^j\}_{n\in\mathbf{Z}} = \{\langle f, \psi_{j,n}\rangle\}_{n\in\mathbf{Z}}$$

The formulas $\forall j \in \mathbf{Z}$

$$\langle f, \psi_{j,k} \rangle = \sum_{n \in \mathbf{Z}} \overline{g_{n-2k}} \langle f, \phi_{j-1,n} \rangle \quad \forall k \in \mathbf{Z}$$

and $\forall j \in \mathbf{Z}$

$$\langle f, \phi_{j,k} \rangle = \sum_{n \in \mathbf{Z}} \overline{h_{n-2k}} \langle f, \phi_{j-1,n} \rangle \quad \forall k \in \mathbf{Z}$$

can then be written as

$$c_k^j = \sum_{n \in \mathbf{Z}} \overline{h_{n-2k}} c_n^{j-1} = (h \# c^{j-1})_k$$

and

$$d_k^j = \sum_{n \in \mathbf{Z}} \overline{g_{n-2k}} c_n^{j-1} = (g \# c^{j-1})_k$$

i.e., $c^j = h \# c^{j-1}$ and $d^j = g \# c^{j-1}$.

These formulas give the effect of the orthonormal basis transformation

$$\{\phi_{j,n}\}_{n \in \mathbf{Z}} \mapsto (\{\phi_{j+1,n}\}_{n \in \mathbf{Z}}, \{\psi_{j+1,n}\}_{n \in \mathbf{Z}})$$

in $\mathbf{V}_j$ on the coefficients of the projection of $f \in L^2(\mathbf{R})$ onto $\mathbf{V}_j$.

In practical applications, this decomposition process is stopped after finitely many iterations, giving a finite sequence of sequences :

$$c^0 \mapsto (d^1, d^2, d^3, \ldots, d^j, c^j)$$

where c^0 is the coefficient sequence of f^0, and the d^j $1 \leq j \leq J$, are the wavelet coefficient sequences representing the difference in information between two consecutive levels of resolution of f^0 , and c^J is the coefficient sequence of the projection of f^0 in $\mathbf{V}_J$.

As the above decomposition operation is equivalent to a cascade of orthonormal basis transformations, the reconstruction operation is the adjoint of the decomposition operation. That is, since $f^{j-1} = f^j + g^j$,

$$c_n^{j-1} = \langle f^{j-1}, \phi_{j-1,n} \rangle = \langle f^j + g^j, \phi_{j-1,n} \rangle$$

$$= \langle \sum_{k\in\mathbf{Z}} c_k^j \phi_{j,k} + \sum_{k\in\mathbf{Z}} d_k^j \psi_{j,k}, \phi_{j-1,n} \rangle$$

$$= \sum_{k\in\mathbf{Z}} c_k^j \langle \phi_{j,k}, \phi_{j-1,n} \rangle + \sum_{k\in\mathbf{Z}} d_k^j \langle \psi_{j,k}, \phi_{j-1,n} \rangle$$

Also, since

$$\phi_{j,k} = \sum_{n\in\mathbf{Z}} h_{n-2k} \phi_{j-1,n}$$

and

$$\psi_{j,k} = \sum_{n\in\mathbf{Z}} g_{n-2k} \phi_{j-1,n}$$

we have

$$c_n^{j-1} = \sum_{k\in\mathbf{Z}} [h_{n-2k} c_k^j + g_{n-2k} d_k^j]$$

That is, $c^{j-1} = h^* \# c^j + g^* \# d^j$, where

$$(a^* \# b)_n = \sum_{m\in\mathbf{Z}} a_{n-2m} b_m$$

and $a^*\#$ is the adjoint of the operator $a\#$.

The above formula suggests a reconstruction algorithm to construct finer resolutions of f from its coarser resolutions and the differences in information between successive resolutions. That is, the coefficient sequence c^{j-1} of f^{j-1} in $\mathbf{V}_{j-1}$ is obtained as follows :

The coefficient sequence c^j of f^{j-1} in $\mathbf{V}_{j-1}$ and the coefficient sequence c^j of f^j in $\mathbf{V}_j$ (representing the difference in information between the $(j-1)$th and jth levels of resolution), are both upsampled by factor 2; the resulting sequences are then convolved with the sequences h and g, respectively, and the resulting convolution products are added up term by term to yield the sequence c^{j-1}. Thus we have the following algorithm :

6.7 The Fast Wavelet Algorithm

This algorithm consists of two sub-algorithms.

1. Wavelet Decomposition Algorithm :

Given by the formulas

$$c_k^j = \sum_{n\in\mathbf{Z}} \overline{h_{n-2k}} c_n^{j-1} \text{ and } d_k^j = \sum_{n\in\mathbf{Z}} \overline{g_{n-2k}} c_n^{j-1}$$

or $c^j = h\#c^{j-1}$ and $d^j = g\#d^{j-1}$, and

2. Wavelet Reconstruction Algorithm :

Given by the formula

$$c_n^{j-1} = \sum_{k\in\mathbf{Z}} [h_{n-2k} c_k^j + g_{n-2k} d_k^j]$$

or $c^j = h^*\#c^j + g^*\#d^j$.

If $a = \{a_n\}_{n\in\mathbf{Z}}$ is any sequence, define by

$$\mathbf{Z}(a) = A(z) = \sum_{n\in\mathbf{Z}} a_n z^n$$

its Z-transform, which is a formal series (i.e., without regard to convergence issues). If $b = \{b_n\}_{n\in\mathbf{Z}}$ is any other sequence, then it is easily verified that

$$\mathbf{Z}(aob) = \sum_{n\in\mathbf{Z}} (\sum_{m\in\mathbf{Z}} a_{n-m} b_m) z^n = (\sum_{n\in\mathbf{Z}} a_n z^n)(\sum_{n\in\mathbf{Z}} b_n z^n) = A(z)B(z)$$

where the product is evaluated term-by-term in the formal algebraic sense.

For any sequence $a = \{a_n\}_{n\in\mathbf{Z}}$, defined

$$\bar{A}(z) = \sum_{n\in\mathbf{Z}} \overline{a_{-n}} z^n$$

If $A(z)$ is the Z-transform of the sequence $a = \{a_n\}_{n\in\mathbf{Z}}$ then $\frac{1}{2}[A(z) + A(-z)]$ is the Z-transform of the downsampled sequence $a \downarrow_2 = \{a_{2n}\}_{n\in\mathbf{Z}}$, and $A(z^2)$ is the Z-transform of the upsampled sequence

$$a \uparrow_2 = \{0 \text{ if } n \equiv 1 \pmod 2), a_{n/2} \text{ if } n \equiv 0 \pmod)\}_{n\in\mathbf{Z}}$$

The decomposition formulas can be rewritten as

$$C^{j-1}(z^2) = \frac{1}{2}[\bar{H}(z)C^j(z) + \bar{H}(-z)C^j(-z)]$$

and

$$D^{j-1}(z^2) = \frac{1}{2}[\bar{G}(z)C^j(z) + \bar{G}(-z)C^j(-z)]$$

While the reconstruction formula can be rewritten as

$$C^j(z) = [H(z)C^{j-1}(z^2) + G(z)D^{j-1}(z^2)]$$

where $C^j(z)$ and $D^j(z)$ are the Z-transforms of c^j and d^j, respectively.

Both the decomposition and the reconstruction formulas can be combined into a single identity by substituting the former in the latter as follows

$$C^j(z) = \frac{1}{2}[H(z)\bar{H}(z) + G(z)\bar{G}(z)]C^j(z) + \frac{1}{2}[H(z)\bar{H}(-z) + G(z)\bar{G}(-z)]C^j(-z)$$

This method of decomposing and reconstructing a function (signal) is called a subband filtering scheme and illustrated by the following diagram :

$$\begin{array}{ccccccccc}
 & & \bar{H} & \rightarrow & \downarrow_2 & \rightarrow & c^j\downarrow_2 & \rightarrow & \uparrow_2 \quad H \\
 & \iddots & & & & & & & \ddots \\
c^{j-1} & & & & & & & & \oplus \quad = c^{j-1} \\
 & \ddots & & & & & & & \iddots \\
 & & \bar{G} & \rightarrow & \downarrow_2 & \rightarrow & d^j\downarrow_2 & \rightarrow & \uparrow_2 \quad G
\end{array}$$

The fast wavelet algorithm can be used to modify a signal's characteristics locally in frequency and time without affecting the signal globally. This is extremely important for purposes of filtering and compression of signals.

Chapter 7

Wavelet Pyramidal Decomposition of Images

7.1 Introduction

An image is a compactly supported, real-valued function in $L^2(\mathbf{R}^2)$. In practice, it is discretely sampled and represented by a matrix $c = (c(i,j))_{R\times C}$ with R rows and C columns, with integer entries ranging from 0 to $K-1$, for some positive integer K. The entry positions (i,j) are called *pixels*, and the entries $c(i,j)$ are called the *pixel values*, and represent the image intensity (gray-level value) at the corresponding pixels.

In this chapter, we discuss a two-dimensional version of Mallat's fast wavelet algorithm, in the context of image decomposition and reconstruction. We shall see that the two dimensional fast wavelet algorithm is a *pyramidal* algorithm, i.e., the original image is decomposed into and represented by several images of varying resolutions (scales,) and containing different aspects of the original image's features at various scales. Moreover it has the additional features of optimality of storage and sensitivity to the orientation of the various features of the image. For ease of understanding and illustration of the algorithm's features, we first look at the one dimensional fast wavelet algorithm, again.

7.2 The 1-D Fast Wavelet Algorithm

In the case of Mallat's fast wavelet algorithm, the decomposition formulas are given by

$$s_k^j = \sum_{n \in \mathbf{Z}} \overline{h_{n-2k}} s_n^{j-1}$$

and

$$d_k^j = \sum_{n \in \mathbf{Z}} \overline{g_{n-2k}} s_n^{j-1}$$

or

$$s^j = h \# s^{j-1}$$

and

$$d^j = g \# s^{j-1}$$

and the reconstruction formulas are given by

$$s_n^{j-1} = \sum_{k \in \mathbf{Z}} [h_{n-2k} s_k^j + g_{n-2k} d_k^j]$$

or

$$s^{j-1} = h^* \# s^j + g^* \# d^j$$

Here,

$$\{s_n^0\}_{n \in \mathbf{Z}}$$

is the input sequence, the sequences

$$\{s_n^j\}_{n \in \mathbf{Z}} \text{ and } \{d_n^j\}_{n \in \mathbf{Z}}$$

are the successive approximation and difference sequences, respectively, and the sequences

$$\{h_n\}_{n \in \mathbf{Z}} \text{ and } \{g_n\}_{n \in \mathbf{Z}}$$

are given by

$$h_n = \langle \phi, \phi_{-1,n} \rangle, \; g_n = \langle \psi, \phi_{-1,n} \rangle \; \forall n \in \mathbf{Z}$$

where ϕ and ψ are the (real-valued) scaling function and orthonormal wavelet of the associated multiresolution analysis $\{\mathbf{V}\}_{j \in \mathbf{Z}}$ of $L^2(\mathbf{R})$, respectively (see sec. 7.4.)

Given a sequence

$$s^0 = \{s_n^0\}_{n \in \mathbf{Z}} \in l^2(\mathbf{Z})$$

(which is the one-dimensional equivalent of a discretely sampled image,) it is identified with a function $f \in \mathbf{V}_0$, given by

$$f = \sum_{n \in \mathbf{Z}} s_n^0 \phi_{0n} = \sum_{n \in \mathbf{Z}} s_n^0 \phi(x - n)$$

Thus $s_n^0 = \langle f, \phi_{0,n} \rangle \; \forall n \in \mathbf{Z}$.

The successive (coarser) approximations of f are given by the projections $\mathbf{P}_j f$ onto the subspaces $\mathbf{V}_j$, $j \geq 1$, of $L^2(\mathbf{R})$, where

$$\mathbf{P}_j f = \sum_{n \in \mathbf{Z}} s_n^j \phi_{j,n}, \; s_n^j = \langle f, \phi_{j,n} \rangle$$

The difference in information between two successive coarse approximations is given by the projections $\mathbf{Q}_j f$ onto the subspaces $\mathbf{W}_j$, $j \geq 1$, of $L^2(\mathbf{R})$, where

$$\mathbf{Q}_j f = \sum_{n \in \mathbf{Z}} d_n^j \psi_{j,n}, \; d_n^j = \langle f, \psi_{j,n} \rangle$$

If the size of the input sequence s^0 is $N = 2^M$ (say), then in the fast wavelet algorithm, the sizes of the difference sequence d^j is $2^{-j}N$, $1 \leq j \leq L$, and the size of s^L is $2^{-L}N$. However, in this case the total size of the decomposition is given by

$$2^{-1}N + \ldots + 2^{-L}N + 2^{-L}N = N$$

Thus the size of the decomposition is the same as the original sequence. This is true in the two dimensional case also.

Thus, there is no redundancy of representation in the wavelet decomposition, wherein the difference sequence d^j represents the difference in the information contents of the successive approximation sequences s^{j-1} and s^j, in the sense that there is no overlap or redundancy in representation between the sequences d^j and s^j. This is because the decomposition is with respect to an *orthonormal* basis

$$\{\psi_{j,k}\}_{j,k\in\mathbf{Z}}$$

and

$$\mathbf{V}_{j-1} = \mathbf{V}_j \oplus \mathbf{W}_j \; \forall j \in \mathbf{Z}$$

. Thus the original sequence s^0 is decomposable into two uncorrelated sequences, namely the difference sequence d^1, and the approximation sequence s^1 , both of which are half the size of s^0, when s^0 is a finite sequence. That is, the size of the decomposed representation is equal to the size of the original sequence. In the two-dimensional case, the image is decomposed into four subimages, each of size $\frac{1}{4}$ of that of the original image. To understand Mallat's algorithm for image decomposition and reconstruction, we must discuss it in its proper setting, namely in two dimensions. We therefore briefly describe i) a method (due to P. G. Lemarié,) of generating a multiresolution analysis for $L^2(\mathbf{R}^2)$ from that of $L^2(\mathbf{R})$, by taking a "tensor product of the MRA of $L^2(\mathbf{R})$ with itself", and ii) the corresponding fast wavelet algorithm.

7.3 An MRA of $\mathbf{L^2(R^2)}$

If

$$\{\mathbf{V}_j\}_{j\in\mathbf{Z}}$$

is an MRA of $L^2(\mathbf{R})$, and ϕ and ψ are the (real-valued) scaling function and orthonormal wavelet associated with the MRA, then the tensor product of the subspace $\mathbf{V}_0$ with itself is given by

$$\mathbf{V}_0 \otimes \mathbf{V}_0 = \{f \in L^2(\mathbf{R}^2) : f(x,y) = \sum_{r,s\in\mathbf{Z}} s_{r,s}\phi(x-r)\phi(y-s), \{s_{r,s}\}_{r,s\in\mathbf{Z}} \in l^2(\mathbf{Z}^2)\}$$

$$= \underset{\mathbf{L^2(R^2)}}{\text{closure}}(span(\{\phi_{0,r}(x)\phi_{0,s}(y)\}_{r,s\in\mathbf{Z}})\}$$

Using the inner product on $L^2(\mathbf{R}^2)$, given by

$$\langle f,g\rangle = \int_{R^2}\int f\bar{g}dxdy,$$

and Fubini's theorem, it can be shown that

$$\{\phi_{0,r}(x)\phi_{0,s}(y)\}_{r,s\in\mathbf{Z}}$$

is an orthonormal basis of $\mathbf{V}_0 \otimes \mathbf{V}_0$, if

$$\{\phi_{0,n}\}_{n\in\mathbf{Z}}$$

is an orthonormal basis of $\mathbf{V}_0$. Defining $\mathbf{V}_j \otimes \mathbf{V}_j$ by

$$f(x,y) \in \mathbf{V}_j \otimes \mathbf{V}_j \Leftrightarrow f(2^j x, 2^j y) \in \mathbf{V}_0 \otimes \mathbf{V}_0,$$

it is easy to verify that

$$\{\mathbf{V}_j \otimes \mathbf{V}_j\}_{j\in\mathbf{Z}}$$

is an MRA of $L^2(\mathbf{R}^2)$, in the sense that it satisfies properties i)- vi) of an MRA of $L^2(\mathbf{R})$, appropriately modified for $L^2(\mathbf{R}^2)$.

Since $\mathbf{V}_{-1} = \mathbf{V}_0 \oplus \mathbf{W}_0$, by using the properties of tensor products it can be shown that

$$\mathbf{V}_{-1} \otimes \mathbf{V}_{-1} = (\mathbf{V}_0 \otimes \mathbf{V}_0) \oplus (\mathbf{V}_0 \otimes \mathbf{W}_0) \oplus (\mathbf{W}_0 \otimes \mathbf{V}_0) \oplus (\mathbf{W}_0 \otimes \mathbf{W}_0)$$

The spaces $\mathbf{V}_0 \otimes \mathbf{V}_0$, $\mathbf{V}_0 \otimes \mathbf{W}_0$, $\mathbf{W}_0 \otimes \mathbf{V}_0$, and $\mathbf{W}_0 \otimes \mathbf{W}_0$ have as bases, the sets of functions

$$\{\phi_{0,r}(x)\phi_{0,s}(y)\}_{r,s\in\mathbf{Z}}$$

$$\{\phi_{0,r}(x)\psi_{0,s}(y)\}_{r,s\in\mathbf{Z}}$$

$$\{\psi_{0,r}(x)\phi_{0,s}(y)\}_{r,s\in\mathbf{Z}}$$

and

$$\{\psi_{0,r}(x)\psi_{0,s}(y)\}_{r,s\in\mathbf{Z}}$$

respectively. If we let $\psi^0 = \phi$, and $\psi^1 = \psi$, then for

$$\sigma \in \{0,1\} \times \{0,1\} = \{(\sigma_1, \sigma_2) : \sigma_i = 0 \text{ or } 1;\ i = 1,2\}$$

we define $\psi^\sigma(x,y) = \psi^{\sigma_1}(x)\psi^{\sigma_2}(y)$. Then, the above bases can be rewritten as

$$\{\Psi^{(0,0)}_{0;r,s}\}_{r,s\in\mathbf{Z}}, \{\Psi^{(0,1)}_{0;r,s}\}_{r,s\in\mathbf{Z}}, \{\Psi^{(1,0)}_{0;r,s}\}_{r,s\in\mathbf{Z}} \text{ and } \{\Psi^{(1,1)}_{0;r,s}\}_{r,s\in\mathbf{Z}}$$

respectively. Also letting $\mathbf{V}_0 = \mathbf{W}_0^0$ and $\mathbf{W}_0 = \mathbf{W}_0^1$, we define $\tilde{\mathbf{W}}_0^\sigma = \mathbf{W}_0^{\sigma_1} \otimes \mathbf{W}_0^{\sigma_2}$, for $\sigma \in \{0,1\}^2$. Similarly, for any $j \in \mathbf{Z}$, we have the spaces $\tilde{\mathbf{W}}_j^\sigma = \mathbf{W}_j^{\sigma_1} \otimes \mathbf{W}_j^{\sigma_2}$ spanned by the orthonormal bases

$$\{\psi^\sigma_{j;r,s}\}_{r,s\in\mathbf{Z}}$$

where

$$\psi^\sigma_{j;r,s}(x,y) = \psi^{\sigma_1}_{j,r}(x)\psi^{\sigma_2}_{j,s}(y)$$

Thus we have a direct sum decomposition of $L^2(\mathbf{R}^2)$:

$$L^2(\mathbf{R}^2) = \bigoplus_{j\in\mathbf{Z}} [\bigoplus_{\substack{\sigma\in\{0,1\}^2 \\ \sigma\neq(0,0)}} \tilde{\mathbf{W}}_j^\sigma],$$

and $\{\psi_{j;r,s}^\sigma : j,r,s \in \mathbf{Z};\ \sigma \in \{0,1\}^2\backslash(0,0)\}$ is an orthonormal basis for $L^2(\mathbf{R}^2)$.

7.4 The 2-Dimensional Wavelet Algorithm

The two dimensional fast wavelet transform of a discretely sampled image is computed using the same scheme as used in the one-dimensional case. First, each row of the image array undergoes decomposition, resulting in an image whose horizontal resolution is reduced by a factor of two, and whose scale is doubled. The high-pass (wavelet filter) component of the decomposition characterizes the high frequency information with *horizontal* orientation. Next, the high-pass and low-pass subimages obtained by the row decomposition are each separately filtered columnwise, to obtain four subimages corresponding to low-low-pass, low-high-pass, high-low-pass, and high-high-pass row-column filtering. These correspond to the coefficient arrays of the image with respect to the subbases

$$\{\Psi_{1;r,s}^{(0,0)}\}_{r,s\in\mathbf{Z}}, \{\Psi_{1;r,s}^{(0,1)}\}_{r,s\in\mathbf{Z}}, \{\Psi_{1;r,s}^{(1,0)}\}_{r,s\in\mathbf{Z}} \text{ and } \{\Psi_{1;r,s}^{(1,1)}\}_{r,s\in\mathbf{Z}}$$

respectively. The latter three subbases capture the horizontal, vertical and diagonal (corners) features, respectively, of the image. To highlight this spatial orientation selectivity feature of these subbases, we rename the corresponding "mother" wavelets a little more suggestively:

We let

$$\Psi^{\mathbf{H}}(x,y) = \psi^{(0,1)}(x,y) = \phi(x)\psi(y).$$

$$\mathbf{\Psi^V}(x,y) = \psi^{(1,0)}(x,y) = \psi(x)\phi(y).$$

$$\mathbf{\Psi^D}(x,y) = \psi^{(1,1)}(x,y) = \psi(x)\psi(y).$$

Also, we let

$$\mathbf{\Phi}(x,y) = \psi^{(0,0)}(x,y) = \phi(x)\phi(y).$$

Thus $\mathbf{\Phi}$ is the two dimensional scaling function, and $\mathbf{\Psi^H}$, $\mathbf{\Psi^V}$ and $\mathbf{\Psi^D}$ are the three orthonormal wavelets corresponding to the three orientation selectivities. The corresponding subspaces generated by these functions at various scales are renamed matchingly:

$$\tilde{\mathbf{W}}_j^H = \tilde{\mathbf{W}}_j^{(0,1)}, \tilde{\mathbf{W}}_j^V = \tilde{\mathbf{W}}_j^{(1,0)}, \tilde{\mathbf{W}}_j^D = \tilde{\mathbf{W}}_j^{(1,1)}, \text{ and } \tilde{\mathbf{W}}_j = \tilde{\mathbf{W}}_j^{(0,0)}.$$

After decomposition of the image into a low-pass subimage and three high-pass subimages as described above, the low-pass subimage is again subjected to the row-column filtering operation to obtain a further (coarser) decomposition, and this process is repeated either until the low-pass image has no more interesting features, or a desired number of times.

If f is the representation of the image $\{s_{n,m}^0\}_{n,m\in\mathbf{Z}} \in l^2(\mathbf{Z}^2)$ in $\tilde{\mathbf{V}}_0$; *i.e.*,

$$f = \sum_{n,m\in\mathbf{Z}} s_{n,m}^0 \Phi_{0;n,m},$$

the orthogonal projections of f onto the subspaces $\tilde{\mathbf{V}}_j$, $\tilde{\mathbf{W}}_j^H$, $\tilde{\mathbf{W}}_j^V$ and $\tilde{\mathbf{W}}_j^D$ are given by:

$$\mathbf{P}_j f = \sum_{n,m\in\mathbf{Z}} s_{n,m}^j \Phi_{j;n,m}, \; s_{n,m}^j = \langle f, \Phi_{j;n,m}\rangle,$$

$$\mathbf{Q}_j^H f = \sum_{n,m\in\mathbf{Z}} d_{n,m}^{H;j} \Psi_{j;n,m}^H, \; d_{n,m}^{H;j} = \langle f, \Psi_{j;n,m}^H\rangle,$$

$$\mathbf{Q}_j^V f = \sum_{n,m\in\mathbf{Z}} d_{n,m}^{V;j} \Psi_{j;n,m}^V, \; d_{n,m}^{V;j} = \langle f, \Psi_{j;n,m}^V\rangle,$$

and

$$\mathbf{Q}_j^D f = \sum_{n,m\in\mathbf{Z}} d_{n,m}^{D;j} \Psi_{j;n,m}^D,\ d_{n,m}^{D;j} = \langle f, \Psi_{j;n,m}^D \rangle,$$

respectively. These projections satisfy the relation

$$\mathbf{P}_{j-1} f = \mathbf{P}_j f + \mathbf{Q}_j^H f + \mathbf{Q}_j^V f + \mathbf{Q}_j^D f$$

The reconstruction algorithm is also similar to that of the one-dimensional case. Each subimage at a given scale is subjected to a synthesis by means of the one-dimensional filters, first operating on the columns, and then on the rows of the subimage, and the resulting subimages are added up to obtain the low-pass subimage at the next finer scale. This process is repeated until the original resolution level is reached.

The two-dimensional fast wavelet algorithm can be easily formulated from its one-dimensional counterpart. Corresponding to the operators $h\#$ and $g\#$ of the one-dimensional fast wavelet algorithm, we can define the row and column operator pairs $h_r\#$, $g_r\#$ and $h_c\#$, $g_c\#$, respectively, where each pair operates on the second and first indices of the image array, respectively. That is, for any image at scale j :

$$s^j = \{s_{n,m}^j\}_{n,m\in\mathbf{Z}},$$

defining

$${}_r s^j = h_r \# s^{j-1}$$

by

$$({}_r s^j)_{n,k} = \sum_{m\in\mathbf{Z}} \overline{h_{m-2k}} s_{n,m}^{j-1},$$

$$t^j = g_r \# s^{j-1}$$

by

$$({}_r t^j)_{n,k} = \sum_{m\in\mathbf{Z}} \overline{g_{m-2k}} s_{n,m}^{j-1}$$

and

$$ {}_cs^j = h_c \# s^{j-1} $$

by

$$ ({}_cs^j)_{k,m} = \sum_{n \in \mathbf{Z}} \overline{h_{n-2k}} s^{j-1}_{n,m} $$

$$ {}_ct^j = g_c \# s^{j-1} $$

by

$$ ({}_ct^j)_{k,m} = \sum_{n \in \mathbf{Z}} \overline{g_{n-2k}} s^{j-1}_{n,m}, $$

we can express the subimages at scale j obtained from a low-pass subimage

$$ s^{j-1} = \left\{ s^{j-1}_{n,m} \right\}_{n,m \in \mathbf{Z}} $$

at scale $j-1$ by the following decomposition formulas:

$$ s^j = h_c \# h_r \# s^{j-1} $$

$$ d^{H;j} = g_c \# h_r \# s^{j-1} $$

$$ d^{V;j} = h_c \# g_r \# s^{j-1} $$

$$ d^{D;j} = g_c \# g_r \# s^{j-1} $$

The presence of oriented features are reflected in the magnitudes and profiles of the coefficients of these subimages. Thus orientation selectivity is an additional feature which is a natural fallout of the choice of the MRA of $L^2(\mathbf{R}^2)$, and does not affect the complexity of the algorithm or the storage size. Since the process of decomposition downsamples the subimages by a factor of 2 in each dimension, each of the resulting four subimages requires a fourth of the number of pixels of the previous image for storage. Thus the total size of the representation remains the

same at each decomposition stage. The corresponding reconstruction formulas at each scale are given by:

$$s^{j-1} = h_s^* \# h_r^* \# s^j + h_r^* \# g_s^* \# d^{H;j} + g_r^* \# h_s^* \# d^{V;j} + g_r^* \# g_s^* \# d^{D;j}$$

where the reconstruction operators

$$h_s^* \# h_r^* \#,\ h_r^* \# g_s^* \#,\ g_r^* \# h_s^* \#,\ g_r^* \# g_s^* \#$$

are the adjoints of the decomposition operators

$$h_r \# h_c \#,\ g_c \# h_r \#,\ h_c \# g_r \#,\ g_c \# g_r \#;$$

respectively.

The following schematic diagrams illustrate the decomposition and reconstruction stages of the 2-dimensional fast wavelet algorithm:

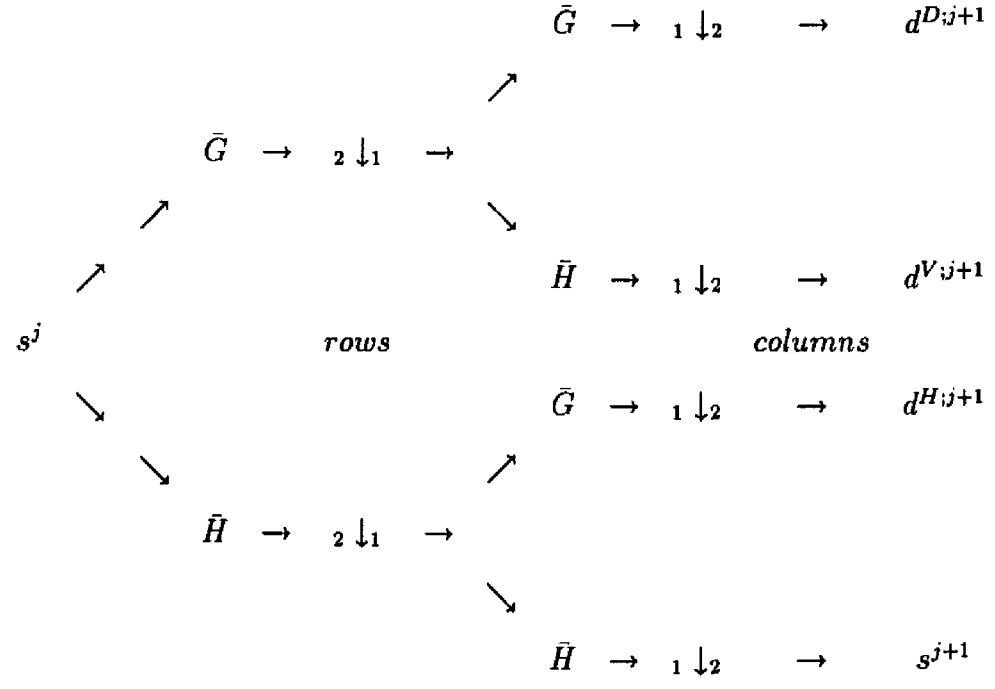


Image decomposition stage: Rows and columns of the low-pass subimage s^j at scale 2^j are convolved with the 1-dimensional CQF filters $\bar{H}$ and $\bar{G}$, to

obtain high-pass subimages $d^{D;j+1}$, $d^{V;j+1}$, $d^{H;j+1}$ and the low pass subimage s^{j+1}, at scale 2^{j+1}. ${}_2\downarrow_1$: Delete every other column.

${}_1\downarrow_2$: Delete every other row.

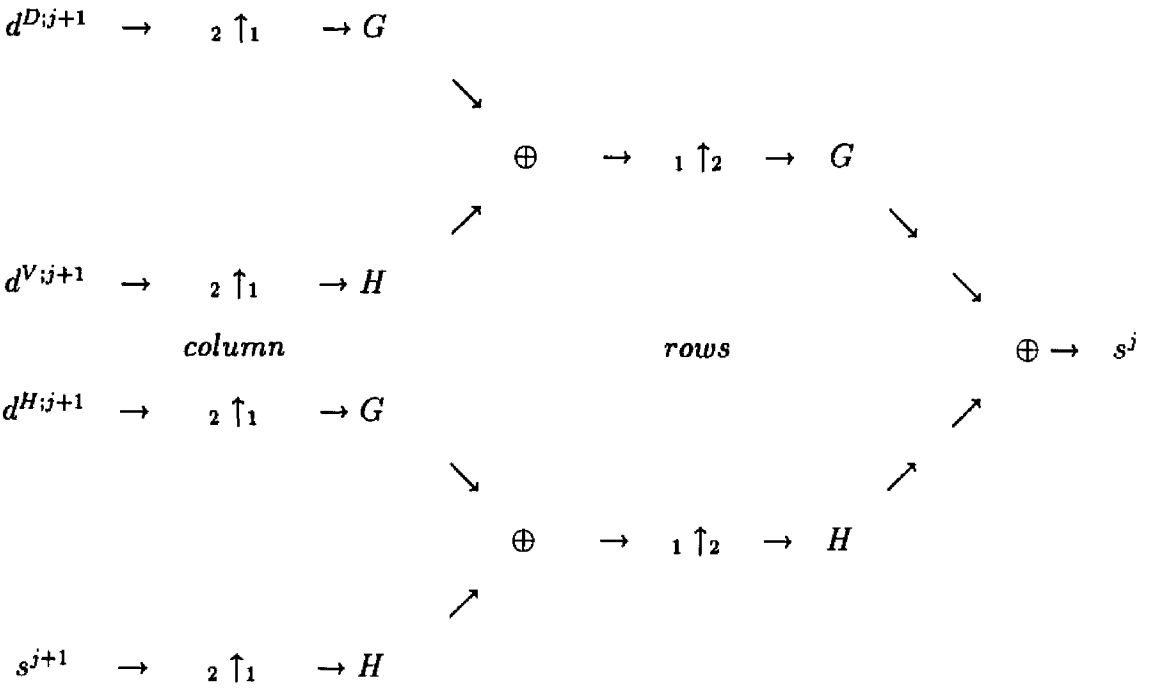


Image reconstruction stage: Rows and columns of the high-pass subimages $d^{D;j+1}$, $d^{V;j+1}$, $d^{H;j+1}$ and the low-pass subimage s^{j+1}, at scale 2^{j+1}, s^j are convolved with the 1-dimensional CQF filters H and G, to obtain the low-pass subimage s^j at scale 2^j. ${}_2\uparrow_1$: Insert a column of zeros between every adjacent pair of columns.

${}_1\uparrow_2$: Insert a row of zeros between every adjacent pair of rows.

$\oplus$: Add.

The two-dimensional fast wavelet algorithm is a pyramidal algorithm, the pyramid generated containing the three high-pass subimages at resolution level 1 at the lowest tier, and the low-pass subimage of the coarsest resolution level at the highest

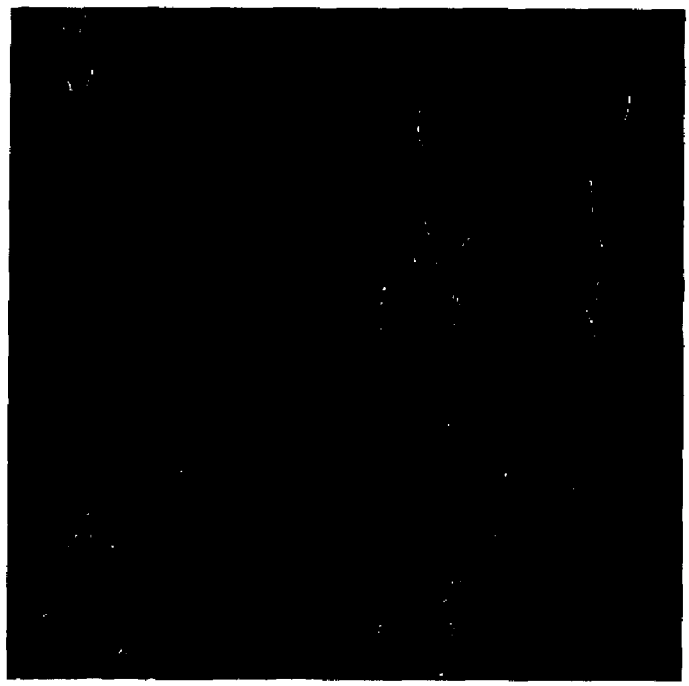


Figure 7.1: The pyramidal decomposition of an image

tier. Each intermediate tier contains three high-pass subimages corresponding to an intermediate resolution level.

Figure 7.1 illustrates the pyramidal decomposition of the Lena image using the 2-dimensional wavelet algorithm with respect to an orthonormal wavelet basis. The figure shows the decomposition of the image at the resolution levels j=1,2,3 into 3 spatially oriented detail images at each level, and a low pass image at level j=3. Figure 7.2 illustrates the original image.

This algorithm is ideally suited for various applications such as image compression, filtering, and multiscale edge detection. To enable fast computation, it is necessary to keep the filter lengths small. Therefore, FIR filters corresponding to compactly supported scaling functions and orthonormal wavelets are used.

Further references: [5, 6, 7, 8, 14, 15, 16, 17].

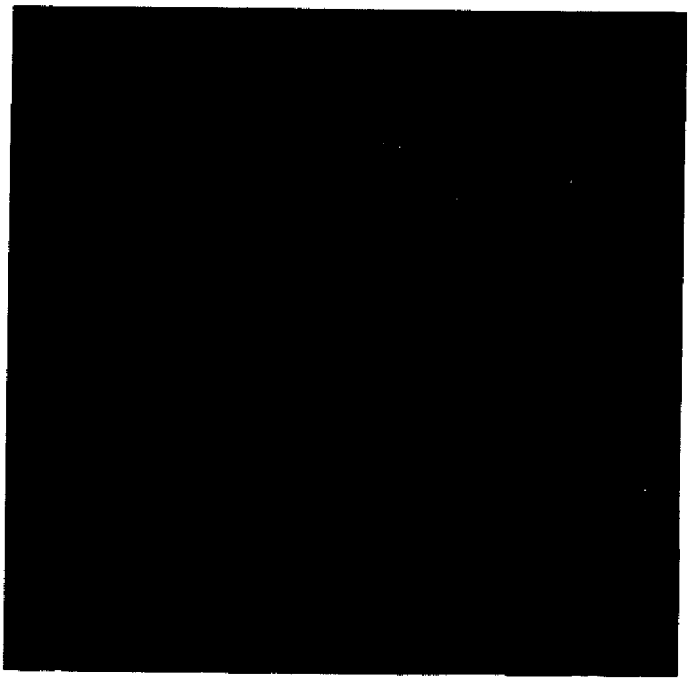


Figure 7.2: Original image

Chapter 8
Object Recognition by Multiresolution Template Matching

8.1 Introduction

The problem addressed here is that of object recognition using template matching. It is assumed that an input image contains finitely many objects from some superset $\mathcal{S}$ of objects. A database of templates is stored, wherein each template τ contains an object from $\mathcal{S}$ and is an $M \times M$ image. The input image $\mathcal{I}$ is to be searched for presence or absence of objects from $\mathcal{S}$, and the types, locations and apparent sizes of those objects present in $\mathcal{I}$ are to be reported, using the template database of objects in $\mathcal{S}$.

Template matching is a fundamental method of detecting the presence or the absence of objects and identifying them in an image. A template is itself an image that contains a feature or an object or a part of a bigger image, and is used to search a given image for the presence or the absence of the contents of the template. This search is carried out by translating the template systematically pixel-by-pixel all over the image, and at each position of the template the closeness of the template to the area covered by it is measured. The location at which the maximum degree of closeness is achieved is declared to be the location of the object detected.

Since in the above described search process a minimum is always attained in the distance between the template and the area covered by it, a predetermined proximity bound is set, as an upper bound for the minimum distance observed. If the minimum observed in the search process is less than this bound, then the object

is declared to be present at the location at which the minimum is attained, else it is declared to be absent.

More precisely, let

$$\mathcal{I} = \{\mathcal{I}(i,j) \mid 0 \leq i,j \leq N-1\}$$

and

$$\tau = \{\tau(i,j) \mid 0 \leq i,j \leq M-1\}$$

be a template of a given object $\mathcal{O}$. A measure of the distance of the template τ from a region of the image $\mathcal{I}$ covered by τ, is given by

$$\mathcal{D}(m,n) = \left[\sum_{j=n}^{M-1+n}\sum_{i=m}^{M-1+m}[\mathcal{I}(i,j) - \tau(i-m, j-n)]^2\right]^{\frac{1}{2}}$$

This is the discrete version of the distance formula

$$\mathcal{D}(u,v) = \left[\int\int_{\mathcal{D}}[\mathcal{I}(x,y) - \tau(x-u, y-v)]^2\, dxdy\right]^{\frac{1}{2}}$$

where, the image $\mathcal{I}$ and the template τ are considered to vary continuously and $\mathcal{D}$ is the domain of definition of the template.

$$\begin{aligned}
\mathcal{D}^2(m,n) &= \sum_{j=n}^{M-1+n}\sum_{i=m}^{M-1+m}\mathcal{I}^2(i,j) + \sum_{j=n}^{M-1+n}\sum_{i=m}^{M-1+m}\tau^2(i-m, j-n) \\
&\quad - 2\sum_{j=n}^{M-1+n}\sum_{i=m}^{M-1+m}\mathcal{I}(i,j)\tau(i-m, j-n) \\
&= \sum_{j=n}^{M-1+n}\sum_{i=m}^{M-1+m}\mathcal{I}^2(i,j) + \sum_{j=n}^{M-1+n}\sum_{i=m}^{M-1+m}\tau^2(i,j) \\
&\quad - 2\sum_{j=n}^{M-1+n}\sum_{i=m}^{M-1+m}\mathcal{I}(i,j)\tau(i-m, j-n)
\end{aligned}$$

The second term on the r.h.s. of the above equation is a constant independent of m and n, being the square of the energy of the template. If it can be assumed

that the energy of the image $\mathcal{I}$ over any window of the size of the template remains approximately constant, then $\mathcal{D}(m,n)$ is minimum when

$$\mathcal{R}_{\mathcal{I},\tau}(m,n) = \sum_{j=n}^{M-1+n} \sum_{i=m}^{M-1+m} \mathcal{I}(i,j)\tau(i-m,j-n)$$

is a maximum.

$\mathcal{R}_{\mathcal{I},\tau}$ is called the *cross-correlation coefficient* of $\mathcal{I}$ and τ at (m,n).

In general however, it is not true that $\sum_{j=n}^{M-1+n} \sum_{i=m}^{M-1+m} \mathcal{I}^2(i,j)$ is approximately constant for all $(m,n) \in \{0,1,\ldots,N-1\} \times \{0,1,\ldots,N-1\}$. If we define the *normalized* cross-correlation coefficient of $\mathcal{I}$ and τ at each point (m,n) by

$$\mathcal{N}_{\mathcal{I},\tau}(m,n) = \frac{\sum_{i-m,j-n=0}^{M-1} \mathcal{I}(i,j)\tau(i-m,j-n)}{\left[\sum_{i-m,j-n=0}^{M-1} \mathcal{I}^2(i,j)\right]^{\frac{1}{2}} \left[\sum_{i,j=0}^{M-1} \tau^2(i,j)\right]^{\frac{1}{2}}},$$

then by the Cauchy-Schwartz inequality,

$$\mathcal{N}_{\mathcal{I},\tau}(m,n) \leq 1$$

with equality iff

$$\mathcal{I}(i,j) = k\tau(i-m,j-n) \quad \forall\, m \leq i \leq M-1+m,\ n \leq j \leq M-1+n,$$

for some scalar $k \neq 0$. As a simple example of a template matching problem, consider the $M \times M$ binary image of an "L-shaped" figure (Figure 8.1) as a template τ.

Given any $N \times N$ $(N > M)$ binary image $\mathcal{I}$, in order to detect the presence (or absence) of the L-shaped object in $\mathcal{I}$, the template τ is translated pixel-by-pixel over the image $\mathcal{I}$, and at each position the correlation coefficient is calculated to check whether it lies within acceptable bounds to declare the presence of the object at that position. Alternatively, the location of the peak value of the correlation

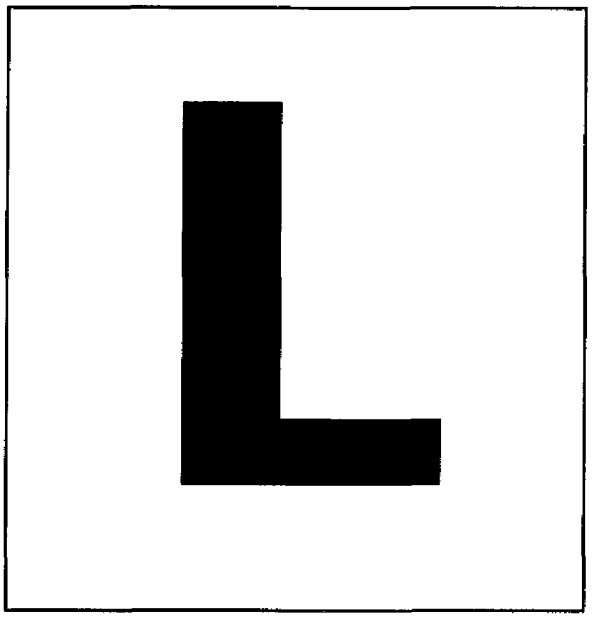


Figure 8.1: Template of an L-shaped object

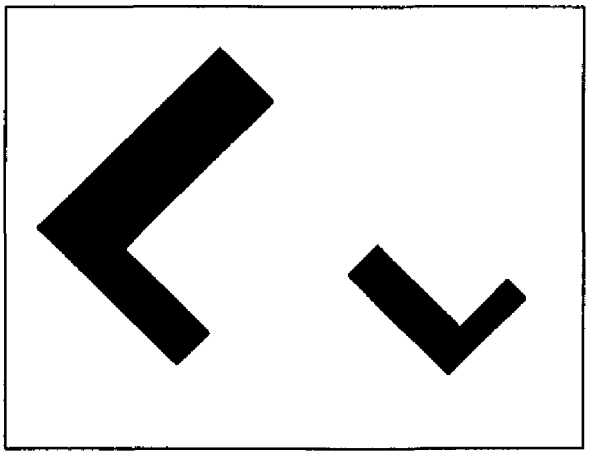


Figure 8.2: Image with scaled 'L's

coefficients computed over the entire image field $\mathcal{I}$ is taken to be the position of location of the object in the image. Clearly, for sufficiently large images $\mathcal{I}$, this method of template matching is computationally intensive. Moreover, scaled and rotated versions of the object in the template occurring in the image (Figure 8.2) may go undetected. In order to accommodate variations in scale and rotation of the object being searched in the image with respect to the object in the template, one would have to perform a range of rotation and scale transformations on the template, and each of these transformed versions of the template correlated with the image at each pixel, leading to an explosive search. Nevertheless, the fundamental necessity of template matching for object, shape and feature recognition in the areas of image analysis, image understanding and robotic and computer vision, make it an indispensable tool. Several efficient implementations of variations of correlation template matching and other alternatives exist with varying degrees and aspects of efficiency, and are constantly being put forward. A brief survey of some of the more important and recent advances in this area is presented in the next section

8.2 Prior Research

The problems of comparing two images for partial or total similarity and of detecting the presence or absence of a given object, shape or feature within a given image, have important applications of far-reaching significance to the areas of pattern recognition, artificial intelligence, and robotic vision, which in turn dictate the advancement of state-of-the-art technology in the industry and the military to a significant extent. As a result of this significance and the complexity of these problems,

a lot of research has gone on in these areas, and several techniques and implementations of these techniques have been proposed towards solving these problems. The key technique to be perfected to solve this problem is that of template matching.

In order to reduce the computational overhead involved in correlation template matching, Wong and Hall [52] proposed a hierarchical template matching search on the pyramidal decomposition of an image, instead of on the image itself. The pyramidal decomposition of an image is a sequence of images obtained by repeatedly low pass filtering and decimating the original image by 2. This sequence of images has lower and lower spatial frequencies and each is a fourth of the previous image in size. Wong and Hall utilize the low spatial frequencies retained in the low resolution images to perform template matching at a considerably reduced computational cost. If the image $\mathcal{I}$ being tested for an object has size $N \times N$, and the template of the object is of size $M \times M$, then the number of test locations i.e. the number of positions at which the correlation coefficient must be computed is $(N - M + 1)^2$. In the hierarchical search, both the image $\mathcal{I}$ and the template τ are decomposed into their respective pyramids consisting of $(L + 1)$; $(0 \ldots L)$ levels each. Thus, at the lowest (coarsest resolution) level L, the number of test locations is reduced to $(N/2^L - M/2^L + 1)^2$, resulting in a savings by a factor of 2^{2L} in the total number of test locations, compared with the highest (finest resolution) level 0.

A sequence of thresholds and a set of decision rules are used to guide the search from a lower resolution level to the next higher resolution level. At the lowest level, the $(N2^{-L} - M2^{-L} + 1)^2$ computations of the correlation coefficients are carried out, and only the (most promising) locations corresponding to values of the correlation coefficients above the threshold value for the level L are then examined for a better and more localized match at the next higher level. Thus, not only is the initial cost of

computation greatly reduced, but the subsequent refinement to the initial estimates are computed only at a few choice locations. The total number of test locations in the hierarchical search on the image pyramid for a template match is given by $K\log(N-M+1)^2+(N2^{-L}-M2^{-L}+1)^2$, where K is a constant. While this method economizes the computations of a correlation based template matching procedure, it does not accommodate for possible scale and rotation variations between the object in the template and its replica in the image. In other words, the hierarchical match has to be performed separately for various possible rotations and scales of the object in the template. Thus there is no savings achieved in the number of transformed versions of the template of each object. In order to uniformly cover even a modest range of scales and rotational angles, a large number of transformed (scaled and rotated) templates needs to be used for each candidate object, and this factor dwarfs the hierarchical match's gain in efficiency, reducing the significance of the savings obtained there.

Anisimov and Gorsky [63] have attempted to alleviate this problem by proposing a hierarchical procedure which takes into account the orientation (rotational) differences in the object in the image with respect to the object in the template. They achieve this by performing the matching operations at each level by rotating the template at each level by a discrete set of angles. The unit angle of rotation generating all the rotations at each level is directly proportional to the scale (and hence inversely related to the resolution). Thus, at the lowest level of resolution, the template rotations are generated by multiples of $\pi/2$. At the jth level of resolution, the unit angle is $\pi/2^{L-j+1}$ and there are 2^{L-j+2} angles of rotation. At the lowest level, all the four rotations of the template are matched using the hierarchical pyramidal technique. Only those angles for which the matching yielded the best fit

are noted as promising orientations of the template, and angles at the next (higher) level in a certain fixed neighborhood of promising angles at the previous level are used for choosing the orientation of the templates at the higher level. That is, the resolution in angular rotation increases at the same rate as the resolution in the template. Again, the multiresolutional approach to orientation matching reduces computation by having fewer different orientations at lower levels and refining only in the neighborhoods of these angles at the previous level for which the match was best. This reduces the number of distinct orientations of a template of each object to 2^L, where L is the number of levels in the image pyramid Since $L = O(\log N)$, the total number of angles is linear in the width of the image. The number of differently oriented templates used in any matching routine is $O(\log N)$. This method does not accommodate scale variations of templates needed to allow for different scaled replicas of the object, in the image. Presumably, one can use discretely scaled versions of various discretely rotated templates and use the above multiresolutional technique to isolate the correct scale and orientation of the template for which the match is best. That is, at the lowest level of the hierarchical search, for each orientation of the template, the match is evaluated at a series of coarse samples of the scale range. The best-fit scales are noted for each angle, and using appropriately defined thresholds, a logarithmic bisection method may be used to choose scales at higher levels of resolution. While these multiresolutional techniques tend to optimize on the number of templates and matches to be performed for the detection of each object, if the number of candidate objects is large, then the computational costs are still prohibitive, and the economy gained in the above methods may not be very significant from a practical point of view.

Rao and Ben-Arie [31] use nonorthogonal expansions of images to obtain template matchings. This method is not a correlation matching technique, but an expansion matching technique, wherein the input image $\mathcal{I}$ is expanded in a nonorthogonal "template-similar" basis generated by translates of the template τ containing the object to be searched for in $\mathcal{I}$. i.e.

$$\mathcal{I}(x,y) = \sum_{l \le k, l \le N} C_{kl}\tau(x-k, y-l)\ x,y \in \{1,\ldots,N\}\ . \tag{8.1}$$

Thus,

$$\mathcal{I}(x,y) = \sum_{1 \le i \le n} \hat{C}_i T_i \tag{8.2}$$

where $n = N^2$ and $T_i = \tau(x-x_i, y-y_i);\ (x_i,y_i) \in \{1,\ldots,N\} \times \{1,\ldots,N\}$. Therefore, $\mathcal{I}_j = \langle \mathcal{I}, T_j \rangle = \sum_{1 \le i \le n} \hat{C}_i \langle T_i, T_j \rangle$.

Thus, $R_{TT}C = \mathcal{I}$ or $C = R_{TT}^{-1}\rangle$ where C is the coefficient set of $\mathcal{I}$ with respect to the template-similar basis $\{T_i\}_{i=1}^{N^2} \equiv \{\tau(x-x_i, y-y_i)\}$ and R_{TT} is the autocorrelation matrix $[R_T T]_{ij} = \langle T_i, T_j \rangle$ of the basis functions.

A template match is registered by the coefficient c_j of the image vector corresponding to the template T_j at position j being large in magnitude. This method yields sharper template match registration peaks than the correlation based method. It is also robust to minor rotation and scale variations between the template and the corresponding object in the image. However, the expansion has to be performed individually for each template tested and for each scale and rotation value. Thus, while its performance is superior to correlation in quality and robustness of matching, computationally it is as arduous as correlation template matching.

8.3 Discussion

In the research described above, a common problem that remains unsolved is the large computational overhead. More precisely, while computational simplifications and speedups have been achieved in actual applications of a template to a given image field, not only is the choice of the template arbitrary, since it is not evident off-hand which template is relevant to the image at hand, but also the correct orientation and scale of the template to be applied are unknown, and are picked arbitrarily. The area of the image over which there is the highest likelihood of a match is also unknown, and as a result, the template has to be systematically moved all over the image and the correlation coefficient computed at each position in order to look for a region of highest likelihood of match. To accommodate scale and orientation variations between the object in the template and its replica in the image (if present), a logarithmic division of the possible range of scales and orientation is performed and the search is economized thus in the scale-orientation space. This kind of search for the correct scale-orientation and translation values of a template often leads to weak and spurious template matches and is computationally expensive, especially when there is a large bank of templates to be tested on an image.

8.4 Outline of Proposed Method

An alternate and novel method is proposed here, that addresses the problems discussed above. In particular, the method proposed here is rotation, scale *and* translation invariant matching procedure, which moreover narrows down the number of templates that need to be matched for search/identification of objects in

an image, by eliminating templates that are not candidates for a successful match. The templates are ordered in a database according to some simple and quantifiable geometric attributes and descriptors of the objects they contain. The input image is preprocessed and geometric attributes and descriptors of objects present in it are extracted, for a choice of appropriate templates for matching, and choice of correct *pose* (scale, rotation and translation) transformations for the chosen templates. These choices made, the templates are accordingly scaled, rotated, and applied to the appropriate site in the image whose location is derived from the geometric descriptors obtained from the preprocessing. An overlap of attributes may result in the application of more than one template to a given site.

To reduce the computation involved, the preprocessing is done on the image at a low resolution and reduced size on its pyramidal decomposition, and the application of the templates is done multiresolutionally, stepping up the resolution (and hence size) of the most promising template from the previous matches at a lower resolution.

That is, the given image is decomposed into an image pyramid of (say) $L + 1$ levels, level L having the smallest size and coarsest resolution, and level 0 having the original image. At level L, the image is preprocessed to obtain geometric attributes and quantitative descriptors of the various objects present. These descriptors are used to pick templates from the template database that have descriptors matching the ones obtained from the image. Each of the chosen templates is scaled, and rotated appropriately, using the descriptors that chose it from the database, to match the scale and orientation of the object in the image. The template is then decomposed into its pyramidal form, and the Lth level of the template pyramid is matched at the corresponding site (obtained from the image descriptors) for a match. If several templates qualify under the same descriptor, then they are all

matched, and the ones with the most promising match are retained for matching again at the next higher level of resolution.

Thus, in this method, by preprocessing the image and obtaining the geometric descriptors of the objects in the image, we know rightaway

1. The templates that are likely to find a match and those that will not find a match
2. The required scaling and orientation factors of each of the chosen templates
3. The location in the image at which the scaled and rotated template should be tested for a close match

The method of assigning and computing geometric attributes and descriptors to the objects in the image and the objects in the templates is by enclosing each object in a minimum area rectangle. The image is preprocessed to obtain contours of objects in it. These contours are enclosed in their convex hulls. These convex hulls are then enclosed in rectangles of least possible areas (minimal enclosing rectangles). The objects in the templates are also treated similarly. The geometric attributes associated with each object are

1. The aspect ratio (breadth/length) of the minimal enclosing rectangle (MER)
2. The length of the rectangle
3. The angle between the rectangle's length and the base of the image
4. The center of the MER

The aspect ratio approximately characterizes the shape of the object enclosed and is a scale-free indicator of the 'oblongness' of the object. This also serves to reduce

the number of candidate templates by eliminating those templates whose aspect ratios do not match with those of the objects in the image.

The ratio of the length of the MER of an object in the image to that of an object with corresponding aspect ratio gives the factor by which the template should be enlarged or reduced to match. The angle between the length of the MER and the base of the image gives the amount by which the template must be rotated for a proper match. The center of the MER of the object in the image indicates where the template MER's center should be applied for a proper match.

Chapter 9

Pose Extraction of Objects in Images

The MER of an object is computed by using an algorithm due to Toussaint [54] on the edge pixels of the object, in $O(m \log m)$ time, where m is the number of edge pixels of the object.

9.1 The Minimal Enclosing Rectangle Algorithm

Given a polygon P, it is possible to enclose it in a rectangle R_P of least possible area in $O(N)$ time, where N is the number of vertices of the polygon P, as demonstrated by the following algorithm due to Toussaint [54]. First, a few observations are in order

1. Given a polygon P, one can bound it in a rectangular bounding box, with its sides parallel to the axes, in $O(N)$ time, where N is the number of vertices of P. This is because it takes $O(N)$ time to compute the maximum and minimum X and Y coordinates of a set of N points.

2. The minimal enclosing rectangle R_P of a polygon P contains the convex hull C_P of P. So it is sufficient to construct the minimum enclosing rectangle of the C_P.

3. Given a polygon P, its convex hull can be constructed in $O(N)$ time, where N is the number of vertices of P.

4. By a theorem of Freeman and Shapira (see [54]), the minimum enclosing rectangle of a convex polygon P has a side containing an edge of P.

The technique for obtaining the minimum enclosing rectangle is called the *Rotating Calipers* method.

Let $v_1, \ldots, v_N$ be the vertices of the convex polygon C, in clockwise order. Consider the bounding box B_i of C, one of whose sides contains the side $\overline{v_{i-1}\ v_i}$. Consider the lines L_1, L_2, L_3 and L_4 obtained by extending infinitely in both directions, the sides of B_i. Let v_1, v_2, v_3 and v_4 be the vertices of C in contact with L_1, L_2, L_3 and L_4 respectively. If $v_{i-1}\ v_i \subset L_1$ (say) choose $v_{i_1} = v_j$.

The lines L_1, L_2, L_3 and L_4 are thought of as rotating calipers, that rotate clockwise preserving the relation $L_1 \perp L_2 \perp L_3 \perp L_4$. The positions of the calipers is restricted to those where one of the four lines contains a side of C (due to remark 4 above).

The computation of the next position of the calipers is done by computing the angles θ_i between L_i and $v_i\ v_{i+1}$; $i = 1, 2, 3, 4$. The smallest nonzero θ_i is chosen to be the angle through which the calipers are rotated. The rotations will be carried out until calipers return to the configuration they started off with. Due to symmetry of the caliper configuration, the total rotation can be limited to $\pi/2$.

Clearly, this algorithm has $O(N)$ complexity in both time and space.

The solution to the problem of finding the minimum enclosing rectangle of a convex polygon is not unique, as is evident in the case of P being an acute angled scalene triangle. But it is possible to obtain all the solutions using the above algorithm.

Given a set of N points, it takes $O(N \log N)$ time to obtain the MER of this set since it takes $O(N \log N)$ time to obtain the convex hull of these points. The

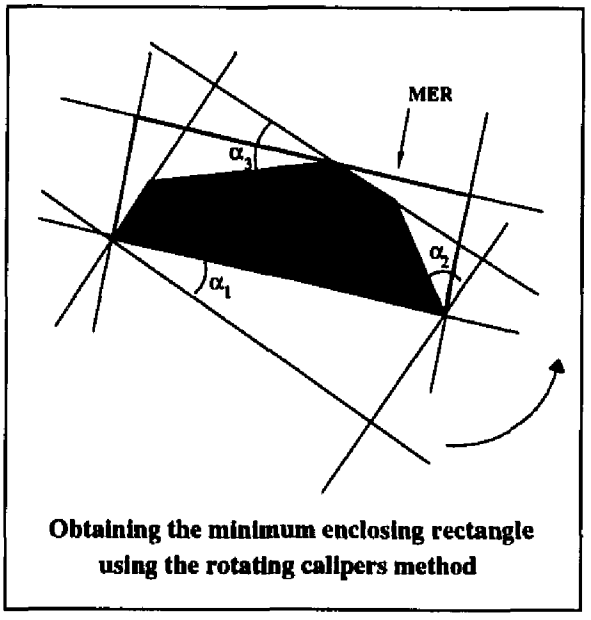


Figure 9.1: Rotating calipers method for computing the MER

convex hull of a set S of N points is the boundary of the intersection of all convex sets containing S. Alternatively, it is the smallest enclosing convex polygon for the set S. The convex hull of a set of N points can be obtained by the Graham's Scan algorithm, which is optimal, and takes $O(N \log N)$ time. The optimality is due to the fact that sorting N points itself takes $O(N \log N)$ time.

9.2 Extraction of Geometric Descriptors using the MER

The above MER algorithm will be used to obtain geometric descriptors of objects present in an input image $\mathcal{I}$.

The templates are classified according to certain geometric attributes of the objects they contain. For each template τ, the object O_τ in τ is associated with its MER. The contour points of the object O_τ are enclosed in an MER, and the aspect ratio $\rho_A(O_\tau)$ of the MER, given by the breadth/length is stored along with the

length $l(O_\tau)$ of the MER, and the center $c(O_\tau) = (c_x(O_\tau), c_y(O_\tau))$ of the MER. The object in the template is assumed to be positioned in such a way that the length of its MER is unity and is parallel to the horizontal side of the template.

The image $\mathcal{I}$ is preprocessed for template matching as follows: A smoothed version of the image is segmented and examined for distinct objects. There are several methods of extracting the outlines/boundaries of objects and labeling them to identify the object to which they belong. Here, we use a seed-fill algorithm to accomplish this. The contour points of distinct objects in $\mathcal{I}$ are labeled according to the object they belong to, and the set of contour points of each object is enclosed in an MER. The aspect ratio $\rho_A(O_\mathcal{I})$ of each of these MER's along with the magnitude of its length $l(O_\mathcal{I})$, center $c(O_\mathcal{I}) = (c_x(O_\mathcal{I}), c_y(O_\mathcal{I}))$ and the angle $\theta(O_\mathcal{I})$ of the length of the MER with the horizontal side of the image are stored in an attribute vector $A(O_\mathcal{I})$ representing the geometric attributes of the object with which the MER is associated.

$$A(O_\mathcal{I}) = [\, \rho_A\,(O_\mathcal{I}),\ l(O_\mathcal{I}),\ \theta(O_\mathcal{I})\ , c_x(O_\mathcal{I}),\ c_y(O_\mathcal{I})\,] \tag{9.1}$$

The first component of the vector A, i.e. ρ_A, gives a scale-free characterization of the 'shape' of the object it is associated with. This is used to locate the right class of templates τ in the template database that are likely to match with the object.

The second component of the vector A, i.e. l, which is the length of the MER of the object is divided by the length of the MER of each template whose aspect ratio matches that of the image object, to obtain the scale factor by which the template must be dilated or contracted in order to match with the object in the image.

The third component of the vector A, i.e., θ, which is the angle of the length of the image object's MER with the horizontal side of the image, is used to rotate the template's MER by θ for aligning the sides of the two MER's for a proper match.

The fourth and fifth component of the vector A, i.e. c_x and c_y, which are the x and y coordinates, respectively, of the center of the MER of the object in the image, are used to apply the center of the candidate template's MER's center onto the object MER's center, for a proper match.

Thus, for each candidate template, the matching procedure is reduced from a search in a 4 dimensional pose (scale-rotation-translation) space to a correlation matching at a single point. Several templates may have the same aspect ratio, in which case, template matching is carried out at the corresponding site in the image, with each of the templates after the appropriately scaling and rotating them. An object may have more than one MER associated with it (with possibly different aspect ratios). In such a case, the largest (or smallest, as desired,) aspect ratio is used to characterize the object, and correspondingly the template corresponding to the object is also stored accordingly in that aspect ratio class.

If an object has more than one MER with largest aspect ratio, then, for each such MER, a corresponding template exists in that aspect ratio class.

Thus, given an image $\mathcal{I}$, the function that maps aspect ratios onto objects in $\mathcal{I}$ is injective, but the mapping from an aspect ratio class in the template database T (see chapter 10) onto an object may be many-to-one. The number of templates that can correspond to a given object is bounded above by the number of sides of the convex hull of the boundary points of the object. This number may be reduced greatly by several polygonal approximation techniques that retain only the most significant vertices of a polygon (see [40, 41, 42, 43]). An optimal multiresolutional

polygonal approximation algorithm for contours of objects is proposed in section 9.4, which is superior in performance to the methods in [40, 41, 42, 43], and is linear in the number of pixels in the object contour.

The preprocessing required on an input image is linear in the size of the image. We, however, lower the computational time by using the pyramidal decomposition of the image and perform matching multiresolutionally.

9.3 Preprocessing of $\mathcal{I}_L$

The input image $\mathcal{I}$ is decomposed in an image pyramid $\mathcal{P}$ with respect to an orthonormal basis of compactly supported wavelets of $L^2(\mathbf{R}^2)$, as described in chapter 7. Thus,

$$\mathcal{I}_0 = \bigoplus_{j=1}^{L} [\mathcal{I}_j^H \oplus \mathcal{I}_j^V \oplus \mathcal{I}_j^D] \bigoplus \mathcal{I}_L \, . \tag{9.2}$$

where $\mathcal{I}_j^H$, $\mathcal{I}_j^V$ and $\mathcal{I}_j^D$ are the detail images with horizontal, vertical and diagonal orientation sensitivity to the images' edge features at scale j and $\mathcal{I}_L$ is the low frequency smoothed and decimated approximation of $\mathcal{I}$ at scale L. If the size of $\mathcal{I}$ is $N \times N$, then the size of $\mathcal{I}_L$ is $N/2^L \times N/2^L$. Thus, there is a 2^{2L}-fold reduction in the number of pixels.

$$\mathcal{I}_0 = \sum_{j=1}^{L} \sum_{k,l \in Z}^{L} [h_{kl}^j \Psi_{kl}^{H,j} + v_{kl}^j \Psi_{kl}^{V,j} + d_{kl}^j \Psi_{kl}^{D,j}] + \sum_{k,l \in Z} [s_{kl}^L \Phi_{kl}] \, . \tag{9.3}$$

The low-pass image $\mathcal{I}_L$ is a smoothed version of the original image $\mathcal{I}$ and has only low spatial frequency features of $\mathcal{I}$ and therefore is less noisy than the original.

$\mathcal{I}_L$ is preprocessed to extract labeled contours of objects in it using segmentation, followed by a seedfill. This results in several sorted lists of pixels, each sorted list corresponding to the ordered boundary points of an object in the image.

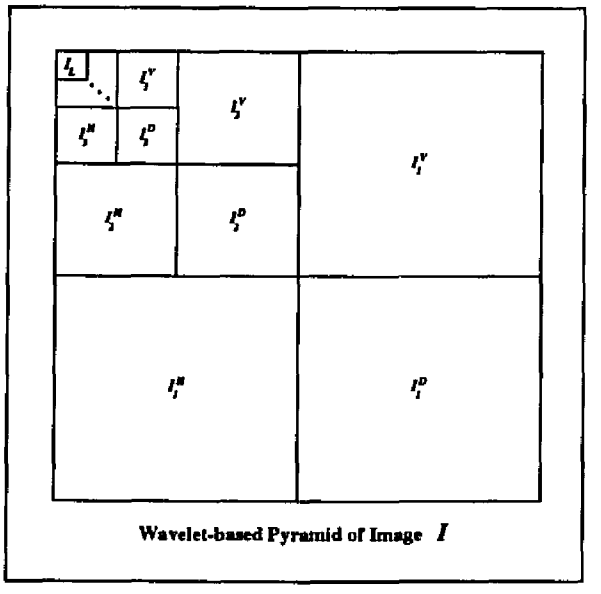


Figure 9.2: Wavelet pyramidal decomposition of an image

The histogram of the image $\mathcal{I}_L$ is smoothed and its prominent modes identified using a multiresolutional technique based on the multiresolution algorithm described in 3.4 (also see [53]). This operation helps identify the most prominent gray levels present in the image. The image is segmented with respect to the gray levels of the peaks of the most prominent modes of the histogram.

The segmented image is then processed using morphological filtering to eliminate noise artifacts and small isolated clumps of pixels. Morphological filtering is done by the well-known operations of opening and closing executed by erosion and dilation of the image. The cleaned image is assumed to have several 'blobs' corresponding to objects in it against a background. This image is further segmented into spatially connected regions or components using a seedfill algorithm.

9.3.1 The Seedfill Algorithm

Let the gray level segmented and filtered image $\hat{\mathcal{I}}_L$ have n gray values $g_1, \ldots, g_n$. If (x, y) is a pixel of $\hat{\mathcal{I}}_L$ denote by $g(x, y)$ the gray value of the pixel (x, y). Thus, for any pixel $(x, y) \in \hat{\mathcal{I}}_L$, $g(x, y) \in \{g_1, \ldots, g_n\}$. The algorithm follows.

$objectnum \leftarrow 1$
$visited(x, y) \leftarrow 0$ (mark all pixels as not visited)
$x \leftarrow 1; y \leftarrow 1$
while $x \leq N$ and $y \leq N$ do
    if $visited(x, y) \neq 1$ then
        push (x, y) onto the stack
        while stack is not empty do
            pop element (i, j) from stack
            label (i, j) in output image as object $objectnum$
            $visited(i, j) \leftarrow 1$
            push 4-neighbors of (i, j) with same gray level onto the stack
        endwhile
        $objectnum \leftarrow objectnum + 1$
    endif
    $x \leftarrow x + 1$
    $y \leftarrow y + 1$
endwhile

The above algorithm starts at a pixel of some gray level and labels all pixels of the same gray value that are spatially adjacent to the original pixel. This process is

repeated until all pixels are visited. Thus, the input image is separated into various regions of similar gray level value.

The border pixels of each set of pixels with the same object number (corresponding to a connected component) are obtained by first choosing a border pixel, and initiating a search for the next border pixel in a counterclockwise (say) direction in its neighborhood.

This method of search obtains a sequence of border pixels whose order in the sequence is the same as their order of occurrence in a counterclockwise traversal of the boundary of the object to which they belong, except for a circular shift. We may subsample this sequence at some rate to obtain a polygonal approximation of the objects' contour. The number of pixels in the approximating polygon's vertex set may be greatly reduced by resorting to algorithms that, given a polygon of N vertices, yield the most significant vertices of the polygon, whose number N' is such that, in general, $N' \ll N$ (for recent results and details see [40, 41, 42, 43] and sec. 9.4).

To the ordered set of vertices of the approximating polygons of the objects obtained as above, the convex hull algorithm is applied to compute the convex hulls of the approximating polygons. The Toussaint algorithm is applied to these convex hulls to obtain the corresponding minimal enclosing rectangles (see Figure 9.3). If more than one MER is available for a polygon, the one with the maximum aspect ratio (breadth/length) is picked and associated with the polygon. The length of the MER is also stored for scaling information required later, at the time of template matching. The center of the MER of each object, in the image coordinate is stored for applying the template MER's center to it in order to obtain a match. The angles of the lengths of the MERs with the base of the image are stored for orientation of

the template to be matched. The aspect ratio is computed and stored for selecting the correct template class.

9.3.2 Preprocessing Complexity

All of the algorithms used for preprocessing $\mathcal{I}_L$ are at most linear in the size of the image.

1. The gray level and connected component segmentation with the intermediate morphological filtering are all linear in the size of the image.

2. The edge pixel detection algorithm is linear in the number of contour pixels of the object in the image, and is much less than the size of the image

3. The polygonal approximation algorithm (see section 9.4) is linear in the number of contour pixels of the object

4. The convex hull algorithm is linear in the number of vertices of the approximating polygons, which is much lesser than the number of contour pixels of the objects, and therefore is very small compared to the size of the image.

5. The Toussaint algorithm is linear in the convex hulls' vertex sets' sizes, (which are smaller than those of the corresponding approximating polygons,) and is again much less than the size of the image

Hence, preprocessing $\mathcal{I}_L$ takes $O(2^{-2L}N^2)$ time.

The preprocessed image $\tilde{\mathcal{I}}_L$ is now ready for template matching using the template database T.

The candidate template is also decomposed in a pyramid of L levels and the template image τ_L at level L is appropriately scaled and rotated and matched at the

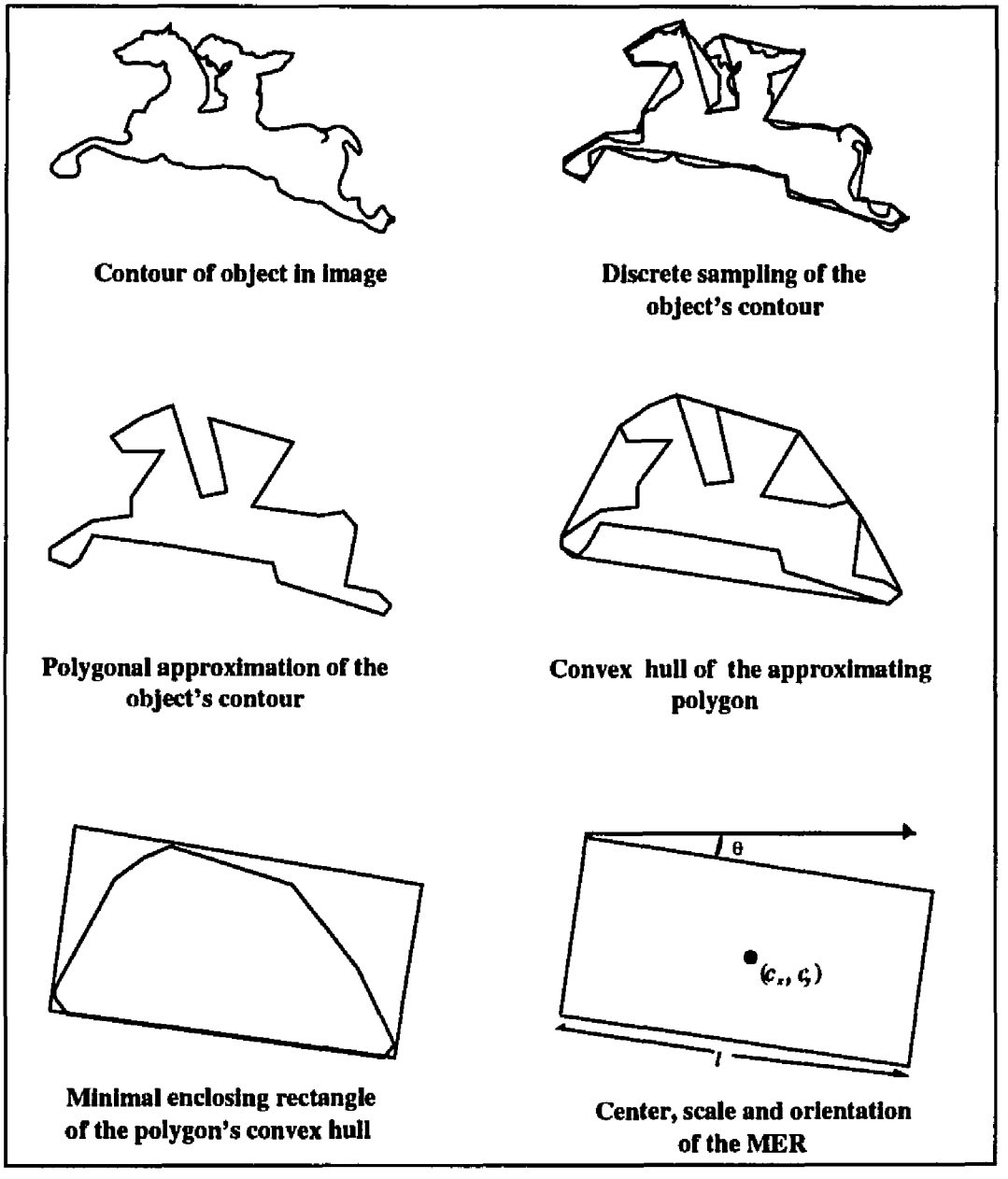


Figure 9.3: Extraction of pose vector from object contour

appropriate site in $\mathcal{I}_L$. If the correlation coefficient is above a threshold, a match is registered at level L. This is performed for all candidate templates, and only those templates yielding correlation coefficients above the prescribed threshold values are matched at the higher level $L-1$. $\mathcal{I}_{L-1}$ and τ_{L-1} are obtained by reconstructing from the images $\mathcal{I}_L, \mathcal{I}_j^H, \mathcal{I}_j^V, \mathcal{I}_j^D$ and the images $\tau_L, \tau_L^H, \tau_L^V, \tau_L^D$ using the wavelet reconstruction algorithm. This is carried out at all sites in $\mathcal{I}_L$ where an object's MER center is located. It is important to note that preprocessing of $\mathcal{I}$ is carried out *only* at level L, in order to obtain the aspect ratios and pose vectors of the objects in $\mathcal{I}$. Matching at a higher level j is performed by reconstructing the low pass images $\mathcal{I}_j$ and τ_j at that level from the four images at the lower level $j+1$ for the image and the template, and the template low pass image τ_j is scaled and rotated and applied to the appropriate site in $\mathcal{I}_j$.

Thus, at each level, template matching is done at a constant number of locations (determined by the number of objects in the image $\mathcal{I}$) and with a fixed number of templates (bounded above by the number of templates in the database T).

9.4 Multiresolutional Polygonal Approximation of Object Contours

We propose here a multiresolutional algorithm for obtaining polygonal approximations of object contours. the algorithm selects the most significant pixels of an object's contour to generate the vertex set of the approximating polygon. Such an algorithm is extremely useful for simplifying shapes of objects. This helps in the economical storage of complex shapes in terms of their most salient features, as well as in the efficient comparison of shapes for purposes of object recognition. Noisy object contours may be cleaned up by polygonal approximations. In the context

of pose extraction of objects in images, the polygonal approximation of object contours drastically reduces the number of points needed to characterize the object shape, thus greatly simplifying and speeding up the convex hull and MER computations. Since the number of pixels in the object contours is typically large, the computation of the MER may suffer from approximation errors that lead to errors in the computation of pose vectors. These errors are due to the large number of edges of very small lengths in comparison to the object's size, and the trigonometric round-off errors in the computation of a large number of very small angles. By using polygonal approximations, the number of vertices needed to characterize the object shape is greatly reduced, resulting in fewer and longer edges. The computation of the MER is made more robust, in the sense that its attitude is governed by the orientation of fewer and longer edges that closely characterize the object contour.

The object contour may be characterized by a Freeman chain-code, which gives the relative angular location θ of each contour pixel with respect to its predecessor in the order of traversal of the contour, as a multiple of $\frac{\pi}{4}$ ($\theta = m\frac{\pi}{4}; m = 0, 1, \ldots, 7$), or, it may be represented by an ordered array of contiguous pixel coordinates. The algorithm described below is adaptable to both representations of object contours. The latter setting of pixel coordinates is used to descrbe the algorithm here, for mathematical convenience.

Let $\{(x_i, y_i)\}_{i=0}^{N-1}$ be a sequence of points representing a simple closed curve Γ in the plane, that corresponds to an object's contour. For purposes of computation we replace the above finite sequence by a doubly infinite periodic sequence $\{(x_i^0, y_i^0)\}_{i \in \mathbf{Z}}$, where $x_j^0 = x_{j(modN)}$ & $y_j^0 = y_{j(modN)}$ $\forall\, j \in \mathbf{Z}$.

For any sequence $a = \{a_i\}_{i\in\mathbf{Z}}$, define the forward and backward difference operators at scale $j \geq 1$ by

$$D_j^+ = S_j^+ - I, \text{ and } D_j^- = I - S_j^-,$$

respectively, where, $S_j^{\pm}a = \{a_{i\pm 2^j}\}_{i\in\mathbf{Z}}$ are the forward and backward shift operators at scale j, and I is the identity operator $Ia = a$.

Thus $D_j^+ = \{(a_{i+2^j} - a_i)\}_{i\in\mathbf{Z}}$, and $D_j^- = \{(a_i - a_{i-2^j})\}_{i\in\mathbf{Z}}$. Clearly, $S_j^{\mp}D_j^{\pm} = D_j^{\mp}$. Denote $x^{j,\pm} = D_j^{\pm}x^0$ and $y^{j,\pm} = D_j^{\pm}y^0$, where $x^0 = \{x_i^0\}_{i\in\mathbf{Z}}$ and $y^0 = \{y_i^0\}_{i\in\mathbf{Z}}$.

If $V_i^0 = (x_i^0, y_i^0)$ and $V^0 = \{V_i^0\}_{i\in\mathbf{Z}}$, define

$$\Theta_j V^0 = \left\{ tan^{-1}\left(\frac{y^{j,+}}{x^{j,+}}\right)\right\}_{i\in\mathbf{Z}} = \left\{\Theta_j V_i^0\right\}_{i\in\mathbf{Z}},$$

and

$$A_j V^0 = \left|\Theta_j V^0 - S_j^-\Theta_j V^0\right| = \left\{\left| tan^{-1}\left(\frac{y^{j,+}}{x^{j,+}}\right) - tan^{-1}\left(\frac{y^{j,-}}{x^{j,-}}\right)\right|\right\}_{i\in\mathbf{Z}} = \left\{A_j V_i^0\right\}_{i\in\mathbf{Z}}.$$

The vector $D_j^+(V_i^0)$ is the displacement of the point $\left(x_{i+2^j}^0, y_{i+2^j}^0\right)$ with respect to the point (x_i^0, y_i^0), and the vector $D_j^-(V_i^0)$ is the displacement of the point (x_i^0, y_i^0) with respect to the point $\left(x_{i-2^j}^0, y_{i-2^j}^0\right)$, the displacements taken along the direction of traversal of the contour.

The sequence $\Theta_j V^0$ gives the angles made by the vectors $\{(x^{j,+}, y^{j,+})\}_{i\in\mathbf{Z}}$ with the x-axis, and the sequence $A_j V^0$ gives the absolute angles subtended by the vector pairs $(x^{j,+}, y^{j,+})$ and $(x^{j,-}, y^{j,-})$, $i \in \mathbf{Z}$. That is, $A_j V_i^0$ is the absolute exterior angle subtended at the point (x_i, y_i) by the chords

$$\overline{\left(x_{(i+2^j)(modN)}, y_{(i+2^j)(modN)}\right)(x_i, y_i)}$$

and

$$\overline{(x_i, y_i)\left(x_{(i-2^j)(modN)}, y_{(i-2^j)(modN)}\right)}$$

of the contour Γ.

For any $1 \leq j < [log_2 N]$, the points $V^0_{i_1}, \ldots, V^0_{i_M}$ that correspond to the local maxima of $A_j V^0$ constitute the vertices of an approximating polygon P^j of Γ. This approximating polygon "captures" the shape of Γ at scale j, while averaging over the finer-scale features of Γ.

The local maxima of the sequences $A_j V^0$, $1 \leq < [log_2 N]$ are used to develop a multiresolution algorithm for polygonal approximations of Γ, by selecting the most significant contour points of Γ. First, a few remarks are in order:

- The local maxima of $A_j V^0$ for large scales j yield polygons that capture the "gross" shape of the curve Γ.
- At large scales, even small local maxima values of $A_j V^0$ may correspond to significant contour points of Γ.
- Maxima at large scales may be the result of the averaging of clusters of smaller scale features, and hence it is necessary to reexamine the neighborhoods of the large scale maxima at finer scales, to search for salient features.
- The strengths of maxima lose their significance with diminishing scale, insofar as indicating the presence of significant vertices is concerned.

The above observations suggest that

1. While selecting the local maxima of the sequences $A_j V^0$, each maximum must be chosen if it is greater than a threshold value t_j.
2. Moreover, the threshold values t_j must increase with decrease in scale.

3. For each selected maximum point V_i^0 at scale j, the sequence $A_{j-1}V^0$ must be searched in the neighborhood $\left\{V_{i+k}^0\right\}_{k=-2^j}^{2^j}$ of the point V_i^0, for maxima at level $j-1$, in order to refine the polygonal approximation obtained at level j.

The number of "kinks" K_Γ in the curve at scale J is estimated by counting the minima in the sequence

$$\left\{\|D_j^+ V_i^0\|\right\} = \left\{\left[\left(x_{i+2^J}^0 - x_i^0\right)^2 + \left(y_{i+2^J}^0 - y_i^0\right)^2\right]^{\frac{1}{2}}\right\}_{i=0}^{N-1}$$

The sampling scale may be chosen by dividing the number of contour pixels N into m equal parts and setting $J = \lfloor log\left(\frac{N}{m}\right)\rfloor$.

The finest scale j_0 for polygon approximation is chosen to be $j_0 = \lfloor log\left(\frac{N}{K_\Gamma}\right)\rfloor$, and the coarsest scale j_1 is set equal to $\lfloor log\left(\frac{N}{k}\right)\rfloor$, where k is the minimum number of sides desired for the approximating polygon. Thus the number of scales s over which the multiresolution approximation is performed is given by $s = j_1 - j_0 + 1 = O\left(log\left(\frac{K_\Gamma}{k}\right)\right)$.

We now present the multiresolutional algorithm for polygonal approximations of an object contour Γ given as a contiguous chain of pixels.

The Algorithm

begin

1. Set $j_0 = \lfloor log\left(\frac{N}{K_\Gamma}\right)\rfloor$, where K_Γ is the number of local minima in the sequence $\left\{\|D_J^+ V_i^0\|\right\}$, and $J = \lfloor log\left(\frac{N}{m}\right)\rfloor$, where $m < N$ and $m = O(N)$

2. Set $j_1 = \lfloor log\left(\frac{N}{k}\right)\rfloor$, where k is the minimum number of sides desired in the approximating polygon.

3. Set the threshold t_j for selecting the maxima of $A_j V^0$, $j_0 \leq j \leq j_1$, at $\frac{2\pi}{K_\Gamma 2^{j-j_0}}$.

4. $j \leftarrow j_1$

5. Choose the indices $i_1^j, \ldots, i_{M_j}^j$ at which $A_j V^0$ attains local maxima that are greater than the threshold t_j.

6. While $j_0 < j$ do

   begin

   For $1 \leq k \leq M_j$ do

   begin

   Obtain the maxima of $A_{j-1} V^0$ in the neighborhood

   $$i_k^j - 2^j \leq i \leq i_k^j + 2^j$$

   Choose the indices

   $$i_{k_1}^{j-1}, \ldots, i_{k_{j-1}}^{j-1}$$

   at which the maxima found above exceed the threshold t_{j-1}. If no maxima are found that are greater than the threshold, then retain the index of the old maximum i_k^j.

   end

   Let the new set of maxima indices be $\{i_1^{j-1}, \cdots, i_{M_{j-1}}^{j-1}$

   $j \leftarrow j - 1$

   end

7. Output all the ordered selected vertices in the above step as the vertices of the approximating polygon

(Note:The vertices selected at each step inherit the order of the subsequence they are extracted from, and hence the polygon's vertex set is already ordered.)

end.

This method of polygonal approximation is superior to those proposed in [40, 41, 42, 43], in that the compression ratio attained, given by the ratio of the number of pixels in the contour of the curve Γ to the number of vertices in the approximating polygon, is much higher in the case of the algorithm presented here, as compared to the methods in [40, 41, 42, 43]. The compression ratios attained in these methods is of the order of 3 to 5, whereas the method described here gives compression ratios as high as 25.

The overall complexity of the multiresolutional algorithm is linear in the number of pixels in the contour of the given curve Γ. As the check for candidacy of each contour point for the approximating polygon's vertex is linear in the number of pixels in the contour, the multiresolution algorithm is optimal.

9.4.1 Template Matching Using Approximating Polygons

Apart from reducing the convex hull and MER computations, and making the MER computations more robust, the above algorithm has another very important use in the object recognition process. It provides a low complexity algorithm for rapidly comparing the contours of an object in an image O and the object in the template τ chosen for comparison.

After a template is chosen for comparison with an image object via the pose vector, the contour of the object in the template can be obtained by methods described earlier in this chapter, or it may even be precomputed and stored in the

template database. The contours of both the image and the template objects are resampled evenly, such that both the resampled contours have the same number N of points. These normalized contours are then periodized to obtain infinite sequences.

These sequences are used to compute the angle sequences $A_j^O V_O^0$ and $A_j^\tau V_\tau^0$, that give the absolute angles subtended by the neighboring vectors $D_j^+ V_O^0$ and $D_j^- V_O^0$, and $D_j^+ V_\tau^0$ and $D_j^- V_\tau^0$, respectively, at scale j. The angle sequences are a scale-orientation-translation independent characterization of the image and template objects' shapes. These angle sequences are searched for maxima that clear the thresholds, and the resulting angle maxima sequences which have a much shorter length than the angle sequences, may now be matched either at a particular scale of interest, or at $O(log\,(max(K_{\Gamma_O}, K_{\Gamma_\tau})/k))$ scales successively or parallely, using a correlation match in the $l^2(\mathbf{Z})$ norm. This is equivalent to a convolution product with wrap-around of two sequences of length $n_j << N$. Thus the overall complexity of this multiresolutional match is $O(nlogn)$, where $n = max_j\{n_j\}$ as the number of scales over which the match is performed is either one or a constant number, depending on the degree of match required.

The degree of match at each level is monitored by a threshold or tolerance value, and further matches are either performed or terminated depending on the degree of match and the required accuracy of the match. This method of template matching using approximating polygons reduces the template matching complexity from being linear in the size of the template to being $O(nlogn)$ in the number of vertices n of the finest chosen approximating polygon of the template object's contour.

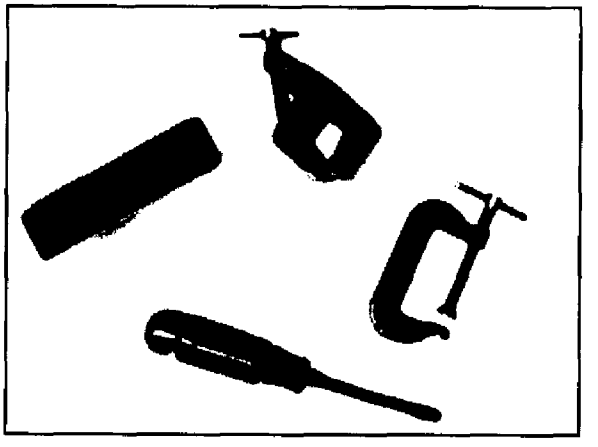


Figure 9.4: Tool Scene

9.5 Implementation of Image Preprocessing

The preprocessing of a given image to extract the pose vectors of the objects contained in a given image is implemented in this chapter on the test image "Tool Scene", containing four objects: a C-clamp, a P-clamp, a stapler and a screwdriver (see Fig. 9.4) against a neutral background.

The image is segmented with respect to its graylevels into a small number of regions, and these regions are relabeled with intensity values which are uniformly distributed in the entire dynamic range of intensity levels. This reduces the number of graylevel regions in the image by assigning the a common grayvalue to several grayvalues sufficiently close to one another. The relabeling of the grayvalues in the segmented image so as to occupy the full dynamic range of graylevel intensity, removes lightish shadows and other clutter from the image background (see Fig. 9.5).

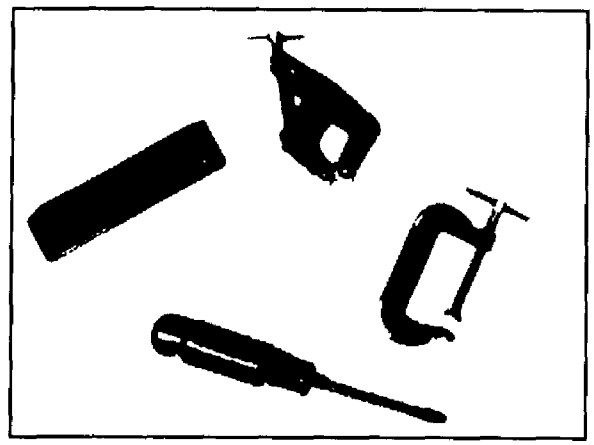


Figure 9.5: Segmented image of Tool Scene

A background fill is performed on the segmented image using the seedfill algorithm subsection 9.3.1., to separate the background from the objects. The background is set to 'white' while the objects are set to 'Black' (see Fig. 9.6) to give a simple binary figure-ground image.

This figure-ground image is treated with a morphological closing operation that lets the black bleed uniformly (dilation), thus closing up narrow gaps and holes, and then retracts the black regions to restore the original size of the objects (erosion.) Closing an image with a compact morphological structuring element smoothes object contours, eliminates small holes in objects and fuses small gaps in objects (see Fig. 9.7). The effect of morphological closing is best seen in the case of the C-clamp.

The closed image is then subjected to a background fill using the seedfill algorithm. This is done to fill up any lacunae of the background formed inside the closed objects, simplifying the geometry of the objects for further processing (see Fig. 9.8).

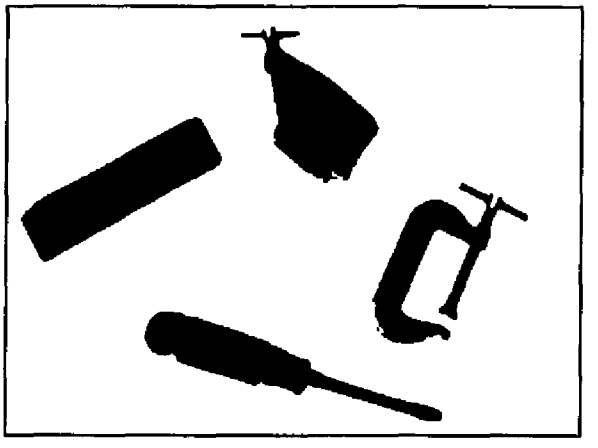


Figure 9.6: Figure-ground image by background seedfill of segmented image

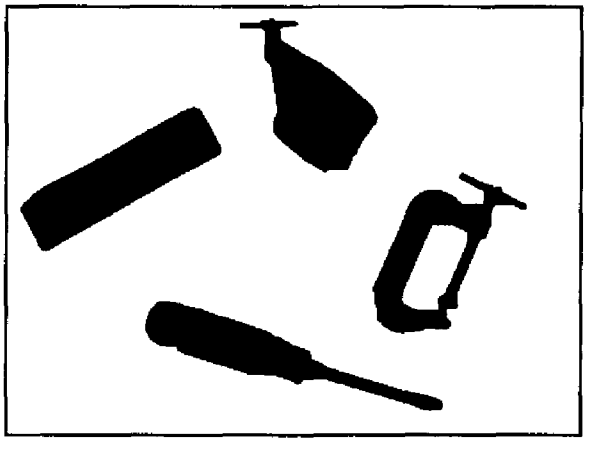


Figure 9.7: Morphological closure of figure-ground image

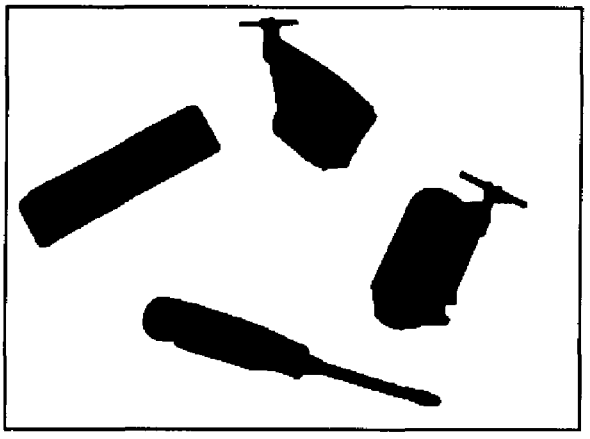


Figure 9.8: Silhouette image from seedfilling of the closed image

This operation ensures that the contour of each object is a simple closed curve. The effect of this operation is best seen in the cases of the C-clamp and the P-clamp. This operation produces the silhouettes of the objects.

The silhouette image is processed by a boundary tracing algorithm (see subsection 9.3.1.,) that obtains the border pixels of each connected object component in a counter-clockwise order along the object's contours (see Fig. 9.9).

The contour image is input to the polygonal approximation algorithm of section 9.4., to approximate each object contour by a polygon that has far fewer vertices than the number of pixels in the object's contours (see Fig. 9.10). This greatly reduces the convex hull and MER computations, and makes the MER computations more robust.

The compression ratios attained for the polygonal approximations are shown in Table 9.1.

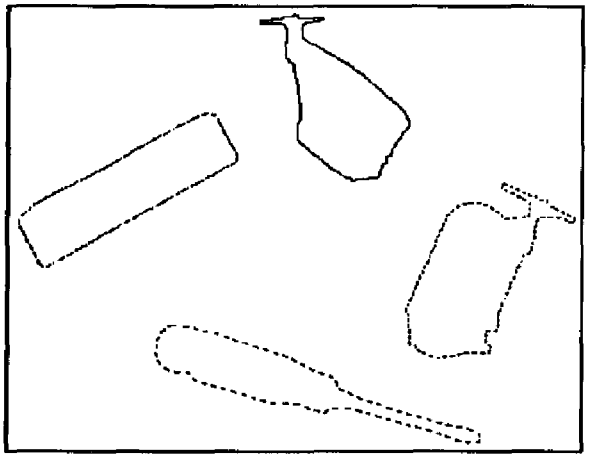


Figure 9.9: Contour image from boundary tracing the silhouette image

Table 9.1: Compression ratios achieved in polgonal approximation of contours

object	# of contour pixels	# vertices in approx. polygon	compression ratio (approx)
C-clamp	605	49	12
P-clamp	504	35	14
Stapler	491	42	12
Screwdriver	691	93	7

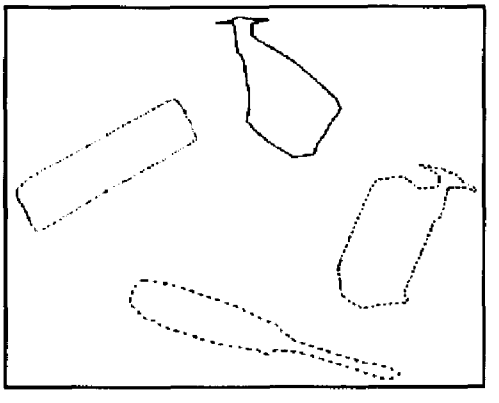


Figure 9.10: Polygonal approximation of contour image

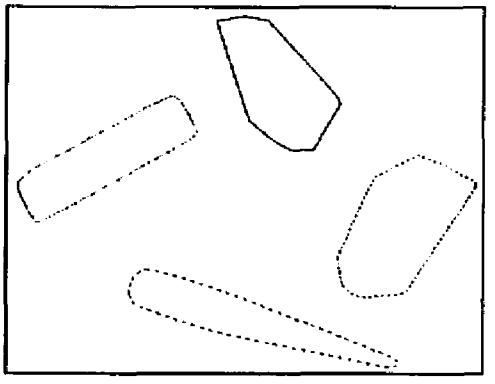


Figure 9.11: Convex hull image obtained from the polygonal image

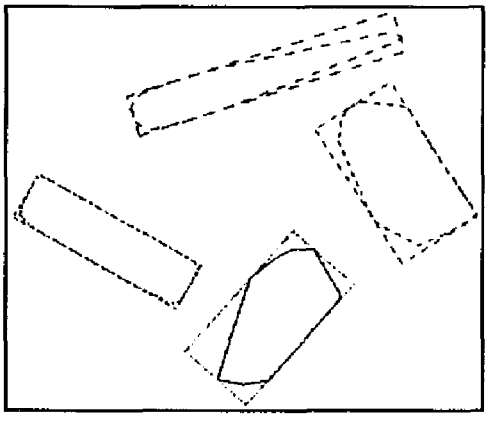
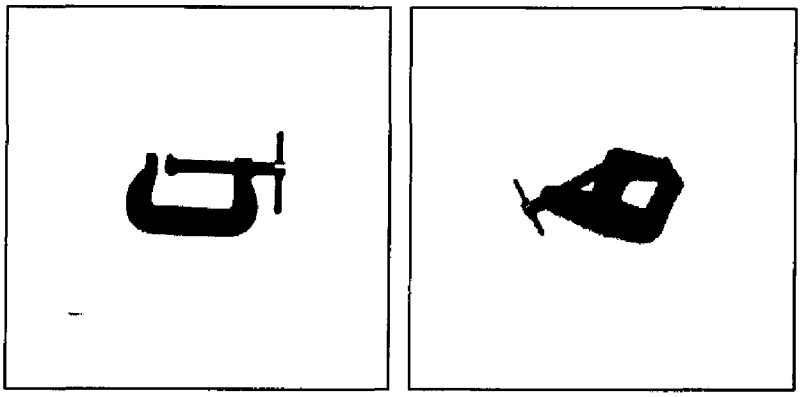
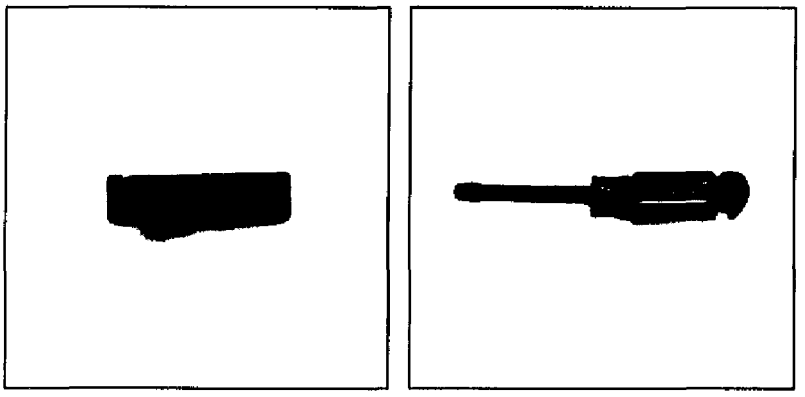


Figure 9.12: The MER image obtained from the convex hull image



(a) Template of C-clamp (b) Template of P-clamp



(c) Template of Stapler (d) Template of Screwdriver

Figure 9.13: Templates corresponding to objects in the tool scene

The polygon image is subjected to the convex hull algorithms to obtain the convex hulls of the various objects (see Fig. 9.11).

The convex hull is then used to compute the minimal enclosing rectangles of the convex hulls using the Toussaint algorithm (see Fig. 9.12).

Figure 9.13 shows the templates of the four objects shown in Tool Scene.

Chapter 10

Guided Template Matching

10.1 The Template Database

If in an application, the number of objects one is interested in identifying is K, then each template in T contains one of the K objects. The MER of each of these K objects is obtained and their aspect ratios $\rho_a(O_i)$; $1 \leq i \leq K$ are computed. If an object O_j has two or more MERs with different aspect ratios, then $\rho_a(O_j)$ is taken to be the maximum (or minimum) of these. The object is rescaled so as to have its associated MER (i.e. the MER with maximum (or minimum) aspect ratio) have its longer side of unit length. The rescaled object is aligned in the template (which is an $M \times M$ image) such that the center of its MER is at the center of the template and the longer side of the MER is of unit length and parallel to the horizontal edges of the template.

If an object O_j has more than one MER of the same (maximum) aspect ratio, then corresponding to each of these MERs there is a template of the object O_j. Thus, more than one template may represent the same object.

Templates in T are classified on the basis of the aspect ratios of the objects they represent. If the K objects have r distinct aspect ratios in all ($r \leq K$), then the database T is divided into r classes $C^1, \ldots, C^r$ parameterized by the aspect ratios $\rho_A^1, \ldots, \rho_A^r$. Each class C^j contains all the templates of the objects O_k for which $\rho_A(O_K) = \rho_A^j$. Within each class C^i, the templates may be ordered arbitrarily or

according to some likelihood weight that tests the templates of objects that are most likely to occur in the input images, first. This choice of ordering is dictated by and may be suited to the application at hand.

A template class is selected for matching with an object in the input image by fitting an MER with the maximum aspect ratio to the object in the image and using the aspect ratio to select the template class. Once a template class is selected for an object in the image, only the templates in that class are used for matching with the object. There are several ways of doing this. One naive approach is to systematically exhaust the whole class by matching each template with the object in question, and retain only those templates that yield a good match (i.e. a high enough cross-correlation coefficient).

This method of searching the class is good enough if the number of templates in each class is small. However, in real applications, each of the classes may contain a large number of templates, in which case, it is inadvisable to conduct the search exhaustively. In such cases, a *guided* search based on previous matches is more economical. To enable such a search, we build a *redirection table* or R-table where the cross correlation coefficients of all the templates of a class are stored in a table.

Denote the crosscorrelation between templates τ_i and τ_j in the same class by X_{ij} i.e.

$$X_{ij} = \frac{\sum_{q=0}^{M-1}\sum_{p=0}^{M-1} \tau_i(p,q)\ \tau_j(p,q)}{\left(\sum_{q=0}^{M-1}\sum_{p=0}^{M-1} \tau_i^2(p,q)\right)^{1/2}\left(\sum_{q=0}^{M-1}\sum_{p=0}^{M-1} \tau_j^2(p,q)\right)^{1/2}}. \tag{10.1}$$

Clearly, $X_{ij} = X_{ji} \leq 1$, and $X_{ii} = 1 \forall i$. This is illustrated in the table below.

To begin with, a template τ_i is arbitrarily selected (or with regard to some likelihood weight) and after being scaled and rotated appropriately it is applied to

Table 10.1: R-table for 4 templates

X_{ij}	τ_1	τ_2	τ_3	τ_4
τ_1	1	X_{12}	X_{13}	X_{14}
τ_2	X_{21}	1	X_{23}	X_{24}
τ_3	X_{31}	X_{32}	1	X_{34}
τ_4	X_{41}	X_{42}	X_{43}	1

the location of the object which selected the class to which the template belongs. The crosscorrelation coefficient obtained from this match is used to look up the R-table in the row corresponding to τ_i. If the computed crosscorrelation is closest to that of τ_j with τ_i from j (closeness is defined by a tolerance bound), then the object is 'redirected' to template τ_j and the match is reperformed with this new template. If $j = i$, (i.e. the crosscorrelation is sufficiently close to 1) then no redirection is done, and a match is said to be found. The redirection sequence is not allowed to contain a template more than once. Thus after a finite number of redirections a match is found, or else the object is declared 'strange'. Alternatively, the best match in the redirection sequence may be elected the closest match, to be considered for further comparison at a higher resolution. If more than one template turns out to be a redirection destination from a current template τ_i, then both these are tested, and the closest match is selected.

10.1.1 Interclass Migration

If the aspect ratios parameterizing the template classes are closely spaced, then errors in computing the aspect ratios of image objects may lead to a wrong choice of classes, in which case the search for a match within the wrongly chosen class may lead to unsatisfactory matches. To provide for this eventuality, we may

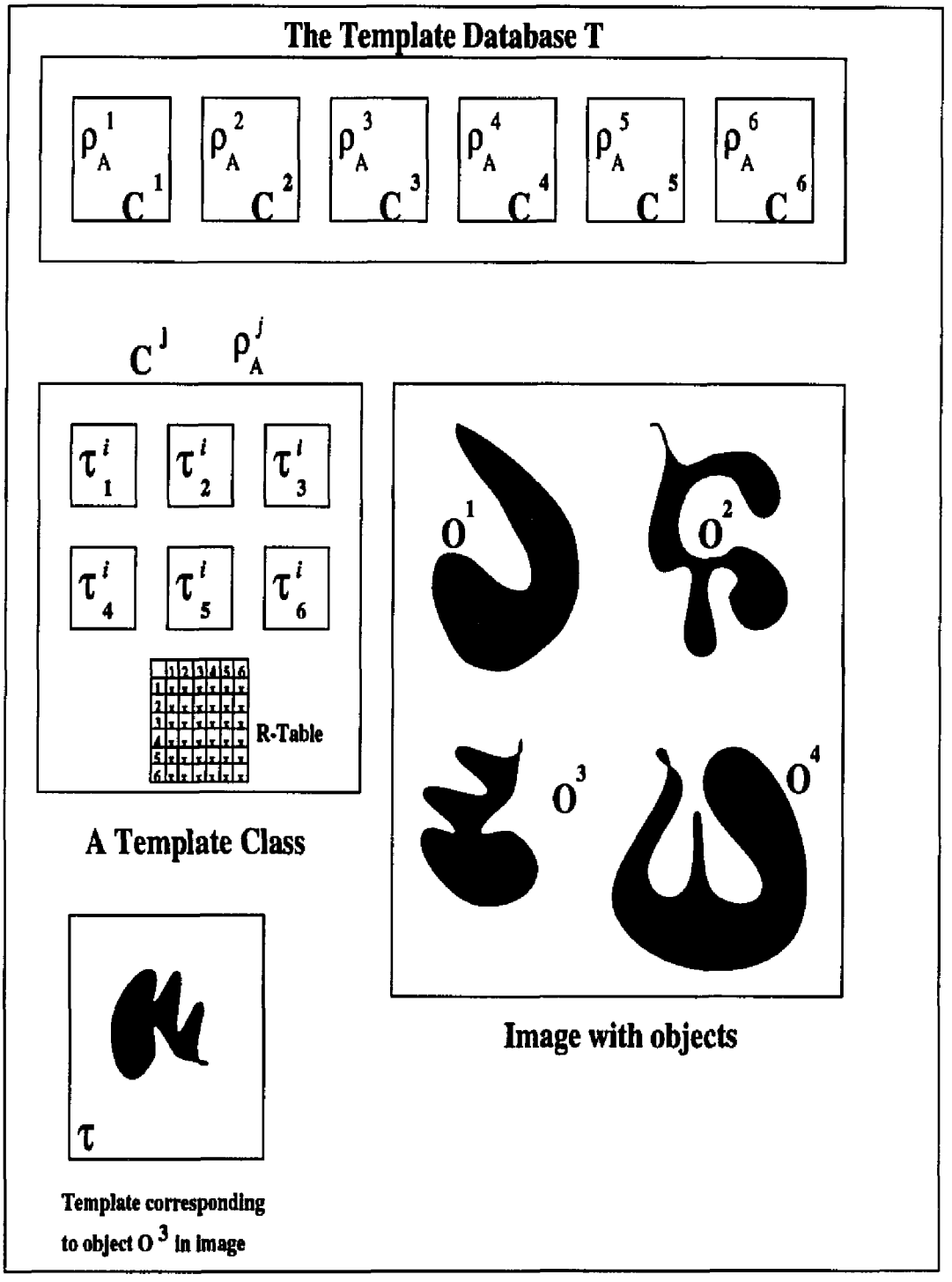


Figure 10.1: The template matching paradigm

imitate the redirection policy employed within a class among templates at the level of classes, and allow a test image object to 'migrate' to another class from the one it is presently assigned. The criterion for choosing the new class is the closeness of the aspect ratio governing this class to the aspect ratio of the migrating object in the image. There are several variations possible on this basic reassignment theme, each of which may be adapted to suit the application.

10.2 Pose-Invariant Template Matching

Let $O_1, \ldots, O_n$ be the objects in the image $\tilde{\mathcal{I}}_L$ with minimum enclosing rectangles $R_1, \ldots, R_n$. Let the aspect ratios of these rectangles be $\rho_A(O_1), \ldots, \rho_A(O_n)$ (not necessarily distinct). Let the pose-vector of object O_i be given by

$$\mathcal{V}_i = (l(O_i), \theta(O_i), c_x(O_i), c_y(O_i))$$

where $l(O_i)$ is the length of R_i, $\theta(O_i)$ is the angle between the length of R_i and the base of $\tilde{\mathcal{I}}_i$, and $c(O_i) = c_x(O_i), c_y(O_i)$ is the center of R_i in the image $\tilde{\mathcal{I}}_L$'s coordinates.

Each object O_i is associated with the template class C_{k_i} of T whose aspect ratio is closest to the aspect ratio $\rho_A(O_i)$ of the object. Two objects may be assigned to the same template class. Let O_i be associated with the template C_{k_i} of T. Thus, $\rho_A(O_i) \approx \rho_A^{k_i}$. Each template τ in C_{k_i} that is computed with the object O_i is transformed using the pose-vector $\mathcal{V}_i$ of O_i. We may assume that the templates are stored in a wavelet image pyramid (as remarked earlier, this does not take any extra space). Thus τ_L is scaled by a factor $l(O_i)$ and rotated by angle $\theta(O_i)$. The rotated and scaled template τ_L' is resized to obtain an $M' \times M'$ window.

The center of this template is applied to the point $c(O_i)$ and the correlation coefficient

$$R_{\mathcal{I}_L\tau_L}(c_x,c_y) = \frac{\sum_{|j-c_y|,|i-c_x|\leq M'/2} \mathcal{I}_L(i,j)\ \tau_L(i-c_x,j-c_y)}{\left(\sum_{|j-c_y|,|i-c_x|\leq \frac{M'}{2}} \mathcal{I}_L^2(i,j)\right)^{1/2}\left(\sum_{|j-c_y|,|i-c_x|\leq \frac{M'}{2}} \tau_L^2(i,j)\right)^{1/2}} \quad (10.2)$$

is computed. Since $\mathcal{I}_L(x,y)=\sum_{i,k} s_{ik}^L\Phi_{ik}^L(x,y)$ and $\tau_L'(x-c_x,y-c_y)=\sum_{i,k} t_{ik}'^L\Phi_{ik}^L(x,y)$. Using the orthonormality of the basis $\{\Phi_{ik}^L(x,y)\}_{i,k\in Z}$ of the subspace V_L of $L^2(\mathbf{R}^2)$, we have

$$\sum_{|j-c_y|,|i-c_x|\leq M'/2} \mathcal{I}_L(i,j)\ \tau_L(i-c_x,j-c_y) = \sum_k\sum_i s_{ik}^L t_{ik}'^L \quad (10.3)$$

and

$$\left(\sum_{|j-c_y|,|i-c_x|\leq \frac{M'}{2}} \mathcal{I}_L^2(i,j)\right)^{1/2} = \left(\sum_k\sum_i |\ s_{ik}^L\ |^2\right)^{1/2}, \quad (10.4)$$

$$\left(\sum_{|j-c_y|,|i-c_x|\leq \frac{M'}{2}} \tau_L'^2(i,j)\right)^{1/2} = \left(\sum_k\sum_i |\ t_{ik}'^L\ |^2\right)^{1/2}.$$

The indices k and i are determined by the range of the template window τ_i' centered at (c_x,c_y) and are finitely many in number. The coefficients s_{ik}^L, t_{ik}^L can be obtained from the scaling function expansion of $\mathcal{I}_L$, τ_L in the pyramidal decomposition and only those coefficients whose associated basis functions' support lies within the template windows domain are chosen. Thus the computational complexity of obtaining the correlation coefficient $R_{\mathcal{I}_L,\tau_L}$ is $O(M'^2)$. Since $M'^2\leq 2^{-2L}N^2$ we have $O(M'^2)=O(2^{-2L}N^2)$ where N^2 is the size of the original image. Thus by choosing L judiciously, the computational cost of template matching can be significantly reduced. The choice of templates selected for matching, within a class may be exhaustive or may follow a guided search based on a redirection table, as discussed

before. For each object O_i, the template of closest match or the set of templates of closest match (depending on the actual implementation preference) is carried over to the next level for matching at a higher resolution, for further confirmation of the identity of the recognized object. The number of levels at which matching is carried out between image and template, depends on the accuracy of match required.

It is important to note that the number of sites at which template matching is done in the low-pass image *does not* depend on the number of pixels in the image, but *only* on the number of objects present in it. This is a significant improvement in computational time over the previously discussed methods.

10.3 Suppressing the Template Background

The template of an object is an $M \times M$ image of the object against some neutral background. This background may contain noise and other extraneous features that reduce the strength of the match. Since we are primarily interested in object recognition, we may suppress the background interference to obtain a pure object-to-object match. This is done by associating a weight function w_τ with each template τ. Since the matching is done multiresolutionally using a pyramidal decomposition, the weight function must also be defined for each level in the pyramid appropriately.

For each template τ, a weight function $w_\tau = w_\tau^0$ is defined such that for each object pixel (x, y) in τ, $w_\tau = 1$, and for each background pixel (x, y) in τ, $w_\tau = 0$. The weight function w_τ^j at the jth level ($j > 0$) of the template pyramid, for the low-pass image $\tau - j$ is defined recursively using the weight function for the low-pass

image at the $j-1$th level as follows

$$w_\tau^j(x,y) = \frac{1}{4}\sum_{i,j=0}^{1} w_\tau^{j-1}(2x+i, 2y+j)$$

i.e., pixel weights at level $j-1$ are averaged over a 2×2 window and assigned to the pixel corresponding to this window at the jth level.

The template τ_j is multiplied by the corresponding weight function w_τ^j at level j to get the weighted template μ_j,where $\mu_j = \mu_j(x,y) = \tau_j(x,y) \cdot w_\tau^j(x,y)$. The image $\mathcal{I}_j$ at level j is multiplied by w_τ^j to obtain the windowed image $\lambda_j = \lambda_j(x,y) = \mathcal{I}_j(x,y) \cdot w_\tau^j(x,y)$.

The appropriately modified crosscorrelation coefficient $\Theta_{\mathcal{I}_j\tau_j}$ at level j is computed by the formula

$$\Theta_{\mathcal{I}_j\tau_j} = \frac{\sum_{|k-c_y^j|\le\frac{M'}{2}} \sum_{|i-c_x^j|\le\frac{M'}{2}} \mathcal{I}_j(i,k)\mu'_j(i-c_x^j, k-c_y^j)}{\left(\sum_{|k-c_y^j|\le\frac{M'}{2}} \sum_{|i-c_x^j|\le\frac{M'}{2}} \lambda_j^2(i,k)\right)^{\frac{1}{2}} \cdot \left(\sum_{|k-c_y^j|\le\frac{M'}{2}} \sum_{|i-c_x^j|\le\frac{M'}{2}} \mu'^2_j(i,k)\right)^{\frac{1}{2}}}$$

where μ'_j is the scaled and rotated version of the weighted template μ_j, transformed to match the scale and orientation of the object in the image with which the match is performed.

10.4 The Object Recognition Algorithm

Input: $\mathcal{I}$ (image of size $N \times N$); T: Template database

1. Decompose $\mathcal{I}$ into an image pyramid (with respect to a wavelet basis) of L levels $\mathcal{I} \rightarrow \{\mathcal{I}_j^H, \mathcal{I}_j^V, \mathcal{I}_j^D, \mathcal{I}_L\}_{j=1}^L$ $\mathcal{I}_0 = \oplus_{j=1}^L [\mathcal{I}_j^H \oplus \mathcal{I}_j^V \oplus \mathcal{I}_j^D] \oplus \mathcal{I}_L$. This takes $O(N^2)$ time.

2. Preprocess the low-pass image $\mathcal{I}_L$ at level L by segmenting and identifying the boundary points of various objects present in $\mathcal{I}_L$, by a seed-fill algorithm.

3. Compute the convex hulls of the boundary points of the objects; this takes $O(\sum_{i=1}^{k} n_i \log n_i)$ time where n_i is the number of border points of object O_i, and k is the number of objects, in $\mathcal{I}_L$. Also $\sum_{i=1}^{k} n_i \ll 2^{-2L}N^2$, so this computational time is very small compared to the image size.

4. Compute the MERs of the convex hulls obtained from the previous step. This takes $O(\sum_{i=1}^{k} m_i$ time, where $m_i \leq n_i$ is the number of points in the convex hull of O_i, and again $\sum m_i \ll 2^{-2L}N^2$.

5. Compute the aspect ratios $\rho_A(O_i)$ and the pose vectors $(l(O_i), c_x(O_i), c_y(O_i), \theta(O_I))$. This takes $O(k)$ time, where k is the number of objects in $\mathcal{I}$.

6. For each object O_i do

   (a) For each template τ in the aspect ratio class $\rho_A(O_i)$, transform the template τ_l by scaling it using $l(O_i)$ and by rotating it by $\theta(O_i)$. (note: each template is stored in a L-level pyramid)

   (b) set $j = L$

   (c) Apply the transformed template τ_j' at level j to the location $(c_x^j(O_i), c_y^j(O_i))$ in $\mathcal{I}_j$ and compute the correlation coefficient. If the correlation coefficient is above the threshold t_j for level j, accept the template as a promising template at level j; else, reject it

   (d) Reconstruct the image and the selected template pyramids by one step from the level j to obtain $\mathcal{I}_{j-1}$ and τ_{j-1} at level $j-1$.

(e) set j to $j-1$; goto step (c)

7. Output the positions, scale, angle and description of objects corresponding to the unrejected templates.

One can associate many kinds of confidence measures describing the degree of match obtained for the various objects in the image, and there are several ways of defining threshold values at various levels of the pyramidal matching to determine whether to accept or reject a match for comparison at the next higher resolution. The choice of these measures is dependent on the application at hand, and what is acceptable (or not acceptable) for a specific application. For instance, often it may not be useful or necessary to match promising templates at all levels, for a given application.

Chapter 11

Merits and Contributions

11.1 Advantages of Orthonormal Wavelet Pyramid Decomposition

The correlation coefficient is an index of the degree of match between an object and image. Minimizing the energy of the difference of the image and template in the region of the template window is equivalent to maximizing the correlation coefficient of the match. Now,

$$\mathcal{I}_j = \mathcal{I}_{j+1} \oplus \mathcal{I}^{H,j+1} \oplus \mathcal{I}^{V,j+1} \oplus \mathcal{I}^{D,j+1} , \tag{11.1}$$

where $\mathcal{I}_j$ is the low pass image of the jth level of the wavelet image pyramid, $\mathcal{I}_{j+1}$ is the low pass image of the $(j+1)$th level of the pyramid, and $\mathcal{I}^{H,j+1}$, $\mathcal{I}^{V,j+1}$ and $\mathcal{I}^{D,j+1}$ are the three directional high-pass images at level $j+1$ of the pyramid.

Similarly, if τ_j is the low pass image of the template at level j of the template pyramid, then

$$\tau_j = \tau_{j+1} \oplus \tau^{H,j+1} \oplus \tau^{V,j+1} \oplus \tau^{D,j+1} . \tag{11.2}$$

Therefore,

$$\begin{aligned} \|\mathcal{I}_j - \tau_j\|^2 &= \|\mathcal{I}_{j+1} - \tau_{j+1}\|^2 + \|\mathcal{I}^{H,j+1} - \tau^{H,j+1}\|^2 \\ &+ \|\mathcal{I}^{V,j+1} - \tau^{V,j+1}\|^2 + \|\mathcal{I}^{D,j+1} - \tau^{D,j+1}\|^2 . \end{aligned} \tag{11.3}$$

Now if the image $\mathcal{I}$ is sufficiently smooth and noise-free (this can be achieved very well by starting off with a smoothed version of the input image, as this will suffice for

most shape recognition algorithms) then most of the image energy is concentrated in the lower frequencies and thus the higher level detail images will contribute very little to the energy of $\mathcal{I}$. This means that $\|\mathcal{I}_{j+1} - \tau_{j+1}\|^2$ does not differ from $\|\mathcal{I}_j - \tau_j\|^2$ significantly and thus good matches can be achieved at low resolutions, and it is not necessary to continue the matching to several higher levels.

This is a spin-off of the fact that the wavelet basis is orthonormal and the approximation and detail spaces are orthogonal to each other (see Chapters 6 and 7).

Another advantage of using a wavelet based pyramid for decomposing and storing the input images and templates is that the pyramid requires no extra storage than the image of which it is a decomposition. This fact also follows from the orthonormality and the resultant decoupling of the scales. This is not true of other pyramidal schemes (see [8]).

Further, the locality of the basis functions in a wavelet basis allows one to modify the image locally without affecting other areas of the image, in both scale and space. This permits selective scale-space filtering and smoothing operations using only the wavelet coefficients of the image. Moreover, the detail images at each resolution can be used to extract edge information about the objects in an image, and there exist efficient methods for doing wavelet based edge detection in images (see [17],) which is outside the scope of this work.

11.2 Novelties and Improvements

The template matching paradigm put forward here has several improvements and new features over existing methods of related function. The most important are enumerated below.

- In this method, the number of locations in an image at which template matching is done depends only on the number of objects detected in the image and not on the size of the image.

- This method identifies the templates in the template database T that are likely to find a match in a given image, thus making the selection of templates deterministic and more efficient. This is a significant improvement over existing methods, since these methods do not specify any criteria for the choice or rejection of a template. In real applications the number of templates in T is large, and hence an efficient method of narrowing down the possible choices significantly affects the computational overhead of the matching procedure.

- In addition to reducing the number of choices of candidate templates, this method also identifies the correct scaling and rotational transformations to be performed on a candidate template, as well as the location in the image at which the match should be computed. This completely eliminates the search in the pose space for the right pose of the template. In other methods, the template pose is methodically or randomly searched in the pose space, involving repeated matching of templates with incrementally corrected poses based on previous matches. This wastes a lot of computations on searching for

the pose alone. Thus the proposed method minimizes the number of matches to be performed for identification.

- The preprocessing of the image in the method proposed reveals several important aspects of the task at hand:
  - the number of objects in the input image is obtained
  - the aspect ratios of each of these objects identify the template classes to be used to match with each of the objects
  - the pose vectors of the objects specify the geometric transformations to be performed on the candidate templates
  - the locations of application of the selected templates is known in the image

  Thus, the size of the task is easily estimated at the preprocessing stage itself, and this information may be used to further economize on computational time, by introducing other application-specific knowledge and criteria about the image at hand or the possible objects it may contain. In other existing methods the number or location or pose of the objects is generally not known *a priori* and thus no estimate can be made as to the size of the task at hand.

- In most methods, the problem of template matching is addressed in the context of a given image and a given template. The choice and sequence/order of templates to be matched is left unaddressed. In the proposed method, the problem of template matching in the context of object recognition is addressed with the assumption that the task of matching is done on an image, with respect to a large bank of candidate templates. This method proposes the

utilization of geometric information regarding objects detected in the image, to judiciously choose and process templates, thus minimizing and guiding the search in template space, while at the same time eliminating search in pose space. This aspect of the method is a novel and significant improvement over other methods of template matching.

- The proposed method uses a compactly supported two-dimensional separable wavelet basis of $L^2(\mathbf{R}^2)$ for the pyramidal decomposition of the input images and the templates. This economizes on the storage space, since the original image and its pyramidal decomposition take the same amount of representation and storage space. This feature makes the storage of all the templates in the database as pyramidal decompositions, affordable. The approximation of an image at any scale j is obtained using wavelet reconstruction, by means of the two- dimensional fast wavelet algorithm on the lower levels of the pyramid. The reconstruction is done only if and when necessary. For sufficiently smoothed input images, the error in the match with a template at a given image location between successive resolution levels of the pyramid is small. So the matching is robust across scales.

- The *overall* complexity of the proposed template matching method described here, *including* the preliminary wavelet pyramidal decomposition of the input image, the preprocessing of the low pass version of the input image, *and* the guided template matching, is KN^2. Here N^2 is the size of the input image and K is a constant that depends on i) the number of templates in the template database, ii) the number of levels chosen in the pyramids, and

iii) the maximum number of recognizable objects in an input image of size $N \times N$.

- The generality of the proposed method makes it pliable, and it can be adapted to suit a large number of applications. Several modifications can be made in the general scheme proposed to suit specific applications without loss of efficiency. For instance, the method propose here can be adapted to scene analysis of aerial images. Key landmarks and other ground based features present over the terrain that is to be identified, may be encoded into the template database as cue objects for recognition of the region. Templates of constellations of simple objects may also be used to produce unique 'thumbprint' configurations that identify regions on land.

11.3 Applications

The template matching paradigm outlined here is useful in several areas of practical applications. The object and shape recognition capabilities make it ideal for use in robotic vision in industrial and commercial settings, wherein a robot or a robotic arm equipped with a camera may go about recognizing and retrieving tools and instruments or sort a heterogeneous collection of objects issuing from a production line or a conveyor belt. The capability to recognize and identify silhouettes and shapes makes it useful in military applications such as target identification and remote surveillance by means of radar, infrared or photo imaging, friend-foe distinction and air-to-ground reconnaissance. Many problems in pattern recognition and machine vision fall within the scope of this paradigm and would benefit by subscribing to its methods.

11.4 Contributions

A novel and general framework for template matching in the context of object and shape recognition is proposed. The novelties and advantages of this framework over other existing methods are i) its capability to minimize the number of templates used on each object in the image ii) the elimination of search in pose space for the correct affine transform on a selected template to obtain the best match with an object in the image iii) the use of an orthonormal wavelet based pyramidal decomposition scheme for reducing the complexity of template matching and preprocessing of the image, and use of a fine to coarse strategy for economical and incremental confirmation of object- template matches iv) a guided search strategy using redirection tables in the template database to obtain better fitting templates based on previous degree of match. v) A novel spin-off of the multiresolution polygonal approximation algorithm proposed for object contours is template matching using polygonal approximation of objects in images and templates.

11.5 Further Research

The shape/object recognition paradigm using template matching proposed here opens up several new vistas of pattern analysis using geometry and image processing to perform intelligent template matching. Currently, work is in progress to recognize partially occluded objects in an image and objects that are only partially included in the image frame. The strategy for accommodating this variation is to generate subtemplates that correspond to object parts and salient features like tell-tale contours etc., and use these on objects that are rejected by the template database as strange objects.

Another variant is contour recognition which is used on the detail images or edge images of images and scale and rotation-free matching of contour segments is very important in recognizing characteristic signatures of complex objects and features. Feature recognition in noisy and textured images using preprocessing and enhancement of global (low frequency) image characteristics and simple cue-features is useful in analyzing scenes and complex images.

Bibliography

[1] D. Ballard and C. Brown. *Computer Vision*. Prentice-Hall, Englewood Cliffs, NJ, 1982.

[2] J.J. Benedetto and M.W. Frazier Ed.. *Wavelets: Mathematics and Applications*. CRC Press, 1994.

[3] G. Bongiovanni, L. Cinque, S. Levialdi and A. Rosenfeld. Image Segmentation by a Multiresolution Approach. *Pattern Recognition* **26**, 12, pp. 1845–1854, 1993.

[4] P.J. Burt and E.H. Adelson. The Laplacian Pyramid as a Compact Image Code. *IEEE Trans. Commun.*. **COM-31**, 4, pp. 532–540, April 1983.

[5] C.K. Chui. *An Introduction to Wavelets*. Academic Press, 1992.

[6] C.K. Chui Ed. *Wavelets: A Tutorial in Theory and Applications*. Academic Press, 1992.

[7] I. Daubechies Orthonormal Bases of Compactly Supported Wavelets. *Commun. Pure & Appl. Math.*. **41**, pp. 909–996, November 1988.

[8] I. Daubechies *Ten Lectures on Wavelets*. Notes for the CBMS conference (Lowell), SIAM 1992.

[9] J.M. Gauch and S.M. Pizer. Multiresolution Analysis of Ridges and Valleys in Grey-Scale Images. *IEEE Trans. Pattern Anal. Machine Intell.*. **PAMI-15**, 6, pp. 635–646, June 1993.

[10] R.C. Gonzalez and P. Wintz. *Digital Image Processing*. 2nd Ed., Addison-Wesley 1987.

[11] R.C. Gonzalez and R.E. Woods. *Digital Image Processing*. Addison-Wesley 1992.

[12] A.K. Jain. *Fundamentals of Digital Image Processing*. Prentice Hall, 1989.

[13] L.M. Lifshitz and S.M. Pizer. A Multiresolution Hierarchical Approach to Image Segmentation based on Intensity Extrema. *IEEE Trans. Pattern Anal. Machine Intell.*. **PAMI-12**, 6, pp. 529–540, June 1990.

[14] S. Mallat. Multiresolution Approximations and Wavelet Orthonormal Bases of $L^2(R)$. *Trans. Amer. Math. Soc.*. **315**, pp. 69–88, 1989.

[15] S. Mallat. Multifrequency Channel Decompositions of Images. *IEEE Trans. Acous., Speech & Signal Proc.*. **ASSP-37**, 12, pp. 2091–2110, December 1989.

[16] S. Mallat and W.L. Hwang. Singularity Detection and Processing with Wavelets. *IEEE Trans. Info. Theory*. **38**, 2, pp. 617–643, March 1992.

[17] S. Mallat and S. Zhong. Characterization of Signals from Multiscale Edges. *IEEE Trans. Pattern Anal. Machine Intel.*. **PAMI-14**, 7, pp. 710–732, July 1992.

[18] L. Prasad, S.S. Iyengar, R.L. Rao and R.L. Kashyap. Fault-Tolerant Sensor Integration using Multiresolution Decomposition. *Physical Review E.* **49**, 4, pp. 3452–3461, April 1994.

[19] L. Prasad, S.S. Iyengar, R.L. Rao and R.L. Kashyap. Fault-Tolerant Integration of Abstract Sensor Estimates using Multiresolution Decomposition. In *Proc. IEEE Conf. Sys., Man & Cyber.*. Le Touquet, France, Oct. 1993

[20] A. Rosenfeld, Ed.. *Multiresolution Image Processing and Analysis*. Springer-Verlag, New York, 1982.

[21] A. Rosenfeld and A.C. Kak. *Digital Picture Processing*. Volume 2, 2nd Edition. Academic Press, 1982.

[22] A. Rosenfeld and M. Thurston. Edge and Curve Detection for Visual Scene Analysis. *IEEE Trans. Computing.* **C-20**, pp. 562–569, 1971.

[23] R.J. Schalkoff. *Digital Image Processing and Computer Vision*. John Wiley and Sons Inc. 1989.

[24] S.L. Tanimoto and T. Pavlidis. A Hierarchical Data Structure for Image Processing," *Comput. Graphics & Image Proc.*. **4**, pp. 104–119, 1975.

[25] J. Crowley. A Representation for Visual Information. Technical Report, CMU-R1-TR-82-7, Robotic Inst, CMU, 1987.

[26] K. Marzullo. Tolerating Failures of Continuous-Valued Sensors. *ACM Trans. on Computer Systems.* **8**, No. 4, pp. 284–304, November 1990.

[27] P. Chew and K. Marzullo. Masking Failures of Multidimensional Sensors. In *Proc. 10th Symposium on Distributed Systems*. Pisa, Italy, Oct. 1991.

[28] L. Prasad, S.S. Iyengar, R.L. Kashyap, and R.N. Madan. Functional Characterization of Sensor Integration in Distributed Sensor Networks. *IEEE Trans. on SMC.* **22**, Sept.-Oct., 1991.

[29] L. Prasad and S.S. Iyengar. A General Computational Framework for Distributed Sensing and Fault-Tolerant Sensor Integration. Submitted for Publication, Dept. of Computer Science TR, October, 1992.

[30] M.J. Quinn. *Designing Efficient Algorithms for Parallel Computers*. McGraw Hill, 1987.

[31] K.R. Rao and J. Ben-Arie. A Novel Approach for Template Matching by Non-Orthogonal Image Expansion. *IEEE Trans. Circ. Sys. Video Technol.*. **3**, 1, pp. 71–84, February 1993.

[32] J. Ben-Arie and K.R. Rao. Optimal Template Matching by Non-Orthogonal Image Expansion Using Restoration. *Int. J. Mach. Vision Appl.*. in press.

[33] R. Haralick and L. Shapiro. *Computer and Robotic Vision*. Addison-Wesley, Reading, MA, 1992.

[34] A. Klinger and C.R. Dyer. Experiments on Picture Representation using Regular Decomposition. *Comput. Vis. Graph. Image Proc.*. **5**, pp. 68–105, 1976.

[35] H.K. Ramapriyan. A Multilevel Approach to Sequential Detection of Pictorial Features. *IEEE Trans. Comput.*. **C-25**, pp. 66-78, January 1978.

[36] A. Rosenfeld and G.J. Vanderbrug. Coarse-fine Template Matching. *IEEE Trans. Sys. Man, Cybern.*. **SMC-7**, pp. 104-107, February 1977.

[37] R.N. Nagel and A. Rosenfeld. Ordered Search Techniques in Template Matching. In *Proc. IEEE*, pp. 242-244, February 1972.

[38] R.O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis*. Wiley, New York, 1973.

[39] A. Rosenfeld and E. Johnston. Angle Detection on Digitized Curves. *IEEE Trans. Comput.*. **22**, pp. 875–878, 1973.

[40] B.K. Ray and K.S. Ray. An Algorithm for Polygonal Approximation of Digitized Curves. *Pattern Recognition Letters*. **13**, pp. 489–496, July 1992.

[41] B.K. Ray and K.S. Ray. Detection of Significant Points and Polygonal Approximation of Digitized Curves. *Pattern Recognition Letters*. **13**, pp. 443–452, July 1992.

[42] D. Sarkar. A Simple Algorithm for Detection of Significant Vertices for Polygonal Approximation of Chain-coded Curves. *Pattern Recognition Letters*. **14**, pp. 959–964, December 1993.

[43] C.H. Teh and R.T. Chin. On the Detection of Dominant Points on Digital Curves. *IEEE Trans. on Patt. Anal. Mach. Intel.*. **11**, 8, 1989.

[44] D.W. Paglieroni, G.E. Ford and E.M. Tsumimoto. The Position-Orientation Masking Approach to Parametric Search for Template Matching. *IEEE Trans. on Patt. Anal. Mach. Intel.*. July 1994 v 16 n 7

[45] K.R. Rao and J. Ben-Arie. Nonorthogonal Image Expansion Related to Optimal Template Matching in Complex Images. *CVGIP Graphical Models and Image Processing*. March 1994 v 56 n 2

[46] V.A. Anisimov and N.D. Gorsky. Fast Hierarchical Matching of an Arbitrarily Oriented Template. *Pattern recognition letters*. February 1993 v 14 n 2

[47] G. Hall and A. Matias. Rotation, Scale and Translation Invariant Template Matching on a transputer network. *Microprocessors and Microsystems*. 1993 v 17 n 6

[48] M. Hasimoto and K. Sumi. High-Speed Template Matching Algorithm Using Information of Contour Points. *Systems and computers in Japan*. 1992 v 23 n 9

[49] S. Karbacher. Associative Object Recognition by Hierarchic Template Matching. *Optical engineering*. December 1990 v 29 n 12

[50] M.C. Cooper. Formal Hierarchical Object Models for Fast Template Matching. *The computer journal*. August 1989 v 32 n 4

[51] E.I. Bykova. Some Methods of Increasing the Stability of Pattern Recognition by Template Matching. *Cybernetics*. September 1988 v 24 n 5.

[52] R.Y. Wong and E.L. Hall. Sequential Hierarchical Scene Matching. *IEEE Trans. Comput.*. **27**, 4, pp. 359–366, 1978.

[53] R.L. Rao. Multiresolution Techniques in Image Processing. Ph.D. Dissertation, Dept. of Computer Science, Louisana State University, 1995.

[54] G.T. Toussaint. Solving Geometric Problems with the 'Rotating Calipers'. In *Proc. IEEE MELECON*, Athens, Greece, 1983.

[55] G.T. Toussaint. On the Complexity of Approximating Polygonal Curves. In *Proc. IASTED, Int. Symposium on Robotics and Automation*. Lugano, Switzerland, 1985.

[56] G.T. Toussaint. The Convex Hull as a tool in Pattern Recognition. In *Proc. AFOSR Workshop in Comm. Theory and Appl.*. Provincetown, MA, September 1978, pp. 43–46.

[57] G.T. Toussaint. Pattern Recognition and Geometric Complexity. In *Proc. 5th Int. Conf. Patt. Recog.*, Miami Beach (1980) pp. 1324–1347.

[58] L. Prasad and S.S. Iyengar. *Introduction to Wavelet Analysis with Applications to Image Processing*, to appear.

[59] E. Edelsbrunner, D.G. Kirkpatrick and R. Seidel. On the Shape of a Set of Points in the Plane. *IEEE Trans. Info. Theory*. **29**, 4, pp. 551–559, July 1983.

[60] D.B. Kirkpatrick and J.D. Radke. A Framework for Computational Morphology. in *Computational Geometry*, G.T. Toussaint, Ed., Elsevier Science Publishers B.V. (North Holland), 1985, pp. 217–248.

[61] S.L. Tanimoto. Template Matching in Pyramids. *Computer Graphics and Image Processing*. **16**, 2, 1981.

[62] V.V. Alexandrov, N.D. Gorsky and S.N. Mysko. Fast Technique for Recursive Scene Matching using Pyramids. *Pattern Recog. Lett.*. **3**, pp. 413–419, 1985.

[63] V.A. Anisimov and N.D. Gorsky. Matching Arbitrary-Oriented Objects using Pyramid Structures. *Information Problems in Automation* LLIAN, Leningrad, 1988.

[64] S.R. Dubois and F.H. Glanz. An Autoregressive Model Approach to Two-Dimensional Shape Recognition. *IEEE Trans. on Patt. Anal. Mach. Intel.*. **8**, 1, pp. 55–66, 1986.

[65] J.G. Dunham. Optimum Uniform Piecewise Linear Approximation of Planar Curves. *IEEE Trans. on Patt. Anal. Mach. Intel.*. **8**, 1, pp. 67–75, 1986.

[66] L. Gupta and M.D. Srinath. Invariant Planar Shape Recognition using Dynamic Alignment. *Pattern Recognition*. **20**, pp. 267–272, 1987.

[67] T. Pavlidis. Algorithm for Shape Analysis of Contours and Waveforms. *IEEE Trans. on Patt. Anal. Mach. Intel.*. **2**, 4, pp. 301–312, 1980.

[68] T. Pavlidis and F. Ali. A Hierarchical Syntactic Shape Analyzer. *IEEE Trans. on Patt. Anal. Mach. Intel.*. **1**, pp. 2–9, 1979.

[69] J. Sklansky and V. Gonzalez. Fast Polygon Approximation of Digitized Curves. *Pattern Recognition*. **12**, pp. 327–331, 1980.

Vita

Lakshman Prasad received a master's degree in Mathematics from the Indian Institute of Technology, Kanpur, India in June 1986. He is currently a graduate research assistant in the Theoretical Division of the Los Alamos National Laboratory. His research interests include wavelet based image analysis & numerical analysis, computational geometry, chaos & dynamical systems.

DOCTORAL EXAMINATION AND DISSERTATION REPORT

Candidate: Lakshman Prasad

Major Field: Computer Science

Title of Dissertation: Multiresolutional Fault-Tolerant Sensor Integration and Object Recognition in Images

Approved:

Major Professor and Chairman

Dean of the Graduate School

EXAMINING COMMITTEE:

Date of Examination:

04/05/95