A Study on Multi-Data Compression Methods using Fluency Functions

（フルーエンシ関数を用いたマルチデータの圧縮手法に関する研究）

by

Yasuhiro OHTAKI, B.E., M.E.

DISSERTATION
Presented to
the Doctoral Degree Program in Engineering
University of Tsukuba

In Partial Fulfillment
of the Requirement
for the Degree
Doctor of Engineering

UNIVERSITY OF TSUKUBA
March 1994

Abstract

When signals from the outer world are dealt with by a computer, the signals are usually digitized by sampling and quantizing operations. Since a set of sampled values often requires huge storage, various data compression techniques have long been the focus of research. Most of these techniques treat the sampled value as original data, and aim to reproduce the discrete value exactly. The sampled values, however, do not always represent the true nature of the original signal. If we can find what is characterizing the true nature, the signal can be represented in a more compact manner by tolerating the reproduction of different value from the sampled one. In this dissertation, several compression methods are proposed, based on this principle. Each method is designed to identify the distinctive features or salient points which characterize the signal and to represent it in terms of the behavior between these points, using mathematical functions. In this study, fluency functions were used, which can represent various functions with different degrees and dimensions. The effectiveness of the methods is demonstrated with electroencephalogram, binary, and grayscale images.

Table of Contents

Introduction

Compression Method for One-Variable Signal

Compression Methods for Two-Variable Signals

List of Figures

Introduction

Chapter 1

Introduction

Human beings are constantly receiving various kinds of information from the outer world, and processing these in order to determine actions to take. Considering its physical limitations and neuron signal transfer speed, the brain can be said to be an information processing mechanism with very high efficiency. Even using the latest computer technology, it is still impossible to realize a comparable information processing machine. The realization of such a machine is a never-ending quest in the field of computer science. There should be some way to represent the signal in a more compact manner.

When we use a computer to deal with various signals, one of the most fundamental problems is how to represent these signals. Sampling and quantizing is the most primitive method to convert the signal into digital discrete values. However, the data usually require very large storage space. One effective solution is to represent the signal in a more economical manner, or in other word, to compress the signal. Compression methods for various categories of object have long been studied, and many techniques have been developed for acoustics, still images, movies, and so on.

It may be thought that the requirement for data compression has been reduced by the development of large volume storage devices. However, this is not what has happened. The improvement of computer power and large storage devices has made it possible to handle much larger data, such as full color movies. The bandwidth of the communication line, on the other hand, is a finite resource, and the opportunity of data transmission has been increasing rapidly according to the growth of computer networks. Though we are able to implement a compression algorithm with large computational complexity, a new algorithm which achieves better results is required.

In these traditional compression techniques, we can find a common principle. The sampled value is treated as the original signal and an exact reproduction of the value is attempted. These *lossless* compression methods are very important in many fields.

However, when dealing with signals from nature, the sampled information is not exactly as that found in nature. In these cases, *lossy* compression techniques may be enough to represent the true nature of the signal. The standard still image compression scheme JPEG is one example of lossy compression technique. The idea of lossy compression is that we don't have to keep all data if we have significant data.

The compression method proposed in this dissertation can be classified as one of the lossy compression techniques. The essential characteristic of the data compression method proposed here is that it "extracts and represents the feature" of the signal. This will derive omre effective results and compact algorithms.

This dissertation is a study of compression methods for various kinds of signal. Though each compression method deals with signals which have different dimensions, they are all based on a common principle: the extraction of "feature points" which characterize the signal, in order to represent how the signal acts in the interval between each pair of adjacent feature points. Based on the experience that signal in nature usually changes smoothly with some radical changing points, this proposed method is designed to compresses the signal by storing the coefficients of the piecewise polynomial function approximating the signal.

In this dissertation, fluency functions are used to represent the variations in signals. Fluency functions can appropriately approximate the object signal by changing the dimension and order of the function according to the continuous differentiability of the signal. We should note that spline functions are also represented as a fluency function. The effectiveness of the method is verified by applying the compression method to several fields of application.

The remainder of this dissertation is constructed as follows.

In Chapter 2, a compression method for electroencephalography is described as an example of an application for one-variable signals. The electroencephalogram (EEG) is a complex electrical signal which reflects generalized brain activity. In a typical recording session, EEG signals are recorded up to 20 electrodes for about 30 min. Since they need to be stored for many years, the amount of data to be stored becomes very large. In order to reduce storage space, a data compression method is here proposed which makes it possible to store the EEG waveforms digitally while maintaining sufficient quality for diagnosis.

The method is based on several features of an EEG signal. An EEG signal varies in time and can be considered as the sum of a basal wave component with a long cycle and large amplitude, and a superimposed wave component with a very short cycle and small amplitude. Keeping precise information on this superimposed wave component requires huge storage.

The method proposed compresses the data volume by storing the coefficients of the approximation function. The method consists of three stages. In the first stage, the EEG signal is divided into segments. Each segment has a similar basal wave cycle, which is extracted by filtering the lower frequency components of the EEG waveform. In the second stage, each segment is decomposed into two wave components: the basal wave component and the superimposed one. The decomposition is made by roughly approximating the original EEG waveform. The approximation function becomes the basal wave component and the approximation error becomes the superimposed wave component. This enables reduction of the bit-length required to represent the superimposed wave component. In the third stage, the superimposed wave components are approximated. The coefficients of the approximation function for the basal wave and the superimposed one are stored as compressed data.

The performance was verified by applying the method to nine actual EEG signals. A bandwidth of 0.5 ~ 120Hz and a dynamic range of 60dB were achieved by a sampling ratio of 250Hz and an 11bit/sample quantization level. The compression ratio, which varies for the complexity of the waveform, was 0.648 on average. The processing speed for 1024 points, or about 4 seconds, is 7.708 seconds for compression and 0.108 seconds for reconstruction.

In Chapters 3 and 4, compression methods for two-variable signal are described.

In Chapter 3, a system which automatically generates a function-font from given binary (black-and-white) images is described as an application of the compression method for two-variable signals.

Recently, scalable fonts have come to be widely used in the field of Desktop Publishing (DTP). Conventionally, these scalable fonts are created manually. Since there are more than 7,000 Japanese characters, it takes several years to create one typeface. It was in order to avoid this labor that the proposed system, which can generate function-fonts automatically, was developed.

The system generates a function-font from a given binary image by extracting and approximating the boundary points. To preserve the quality of the reconstructed font, the sequence of boundary points is divided into a number of segments. The point where two segments are divided is marked as a feature point.

To extract these feature points appropriately, a multi-stage algorithm is proposed. This algorithm consists of three stages. In the first stage, obvious feature points such as sharp corners and the ends of a straight line, are extracted based on the evaluation of digital curvature. In the second stage, feature points at the junction of two different classes of lines, e.g., the junction of an arc and a straight line, are extracted based on the evaluation of analog curvature. In the third stage, in order to reduce the compressed data volume, redundant joint points which do not affect the quality of the reconstructed image are eliminated.

Finally, boundary points in each interval between adjacent joint points are approximated by fluency function, and the coefficients are stored as compressed data.

The effectiveness was evaluated by applying the method to brush-written Chinese characters, printed characters and drawings.

In Chapter 4, a compression method for grayscale images is described as an another application of the compression method for two-variable signals.

As computational power has increased and made possible the development of large storage devices, an image archival system has come to be seen as a possibility. Such a system needs to be able to store many images and reconstruct them in a short time. A compression method which permits fast reconstruction is therefore required.

The proposed method is constructed by replacing the DCT operation in the JPEG image compression scheme with fluency function approximation. The method compresses the image by approximating the gray-level in 8×8 pixel blocks. The coefficients of the approximate function are then encoded by the Huffman entropy coder. The compression ratio, coding and decoding speed and the quality of the reconstructed image are compared with those of the JPEG image compression scheme.

Chapter 5 presents the conclusions. The scientific and technical results which were obtained through Chapter 2 to Chapter 4 are summarized. Problems beyond the scope of this study are also pointed out.

Compression Method for One-Variable Signal

Chapter 2

Compressing EEG Data

In this chapter, a compression method for electroencephalogram (EEG) has been proposed as an application for the compression method for a one-variable signal.

2.1 Introduction

The electroencephalogram (EEG) is a complex electrical signal which reflects generalized brain activity and provides significant and useful parameters for clinical investigations. In a typical recording session, EEG signals are recorded at up to 20 electrodes for about 30 min. Normal chart paper recording requires about 50m of chart paper. This causes a serious problem in securing enough storage space. Various methods have been proposed to solve this problem.

The microfiche system [104] has the advantage of enormous data reduction, but is problematic in computer analysis. Since computers are frequently used for analysis nowadays, it is appropriate to store the data in a machine readable form, i.e., digitally.

Storing only some parameters obtained through computer analysis [105] [106] also achieves a drastic reduction in storage space, but these parameters show only one aspect of the EEG signal and do not hold sufficient information for diagnosis. For the development of new analysis method in the future, it is desirable to store the waveform itself.

A digital sampling and about 500K bytes of storage per electrode is necessary in the digital storage of an EEG waveform. For a single patient with a full complement of 20 electrodes, storage requirements reach 11M bytes. There are two possible approaches to reduce the required storage. These are (1) the development of a recording device with higher density, and (2) the development of a data compression method. In this paper, a data compression method is proposed for storing the EEG waveforms digitally while maintaining sufficient quality for diagnosis.

2.2 Specifications

This section deals with the specification for the quality of the reconstructed waveform. The reconstructed waveform is required to have sufficient quality for diagnosis. This will be satisfied if the quality is as high as that of the waveform recorded with the conventional EEG measuring and recording system.

The quality is evaluated by the error between the original EEG waveform and the reconstructed waveform, defined as follows:

$$\text{(maximum error)} \triangleq \left| \max_{0 \leq i \leq K-1} (\hat{z}(t_i) - z(t_i)) \right| + \left| \max_{0 \leq i \leq K-1} (z(t_i) - \hat{z}(t_i)) \right|$$

where $\{z(t_i)\}_{i=0}^{K-1}$ and $\{\hat{z}(t_i)\}_{i=0}^{K-1}$ represent the original EEG waveform and the reconstructed one, respectively.

The conventional EEG measuring equipment has a sensitivity of 1μV and demands that an error of more than 3μV does not occur more than once a second. The conventional recording equipment is an FM modulation analog data recorder and has an S/N ratio of 50dB(p-p/rms) and a dynamic range of 60dB. From the relation between the S/N ratio and dynamic range, the tolerance error of the recording equipment becomes about $2\mu\mathrm{V}_{\mathrm{p-p}}$. The specification for the quality required for the reconstructed EEG waveform is summarized in Table 2.1.

Table 2.1: Required quality for the reconstructed waveform

bandwidth	$0.5 \sim 120$ Hz
sensitivity	$1\mu V$
dynamic range	60dB
tolerance error	$2\mu V_{p-p}$

2.3 Compression and Reconstruction Method

2.3.1 Design policy

The EEG waveforms are compressed after converting into descrete values.

The specification of a bandwidth of 0.5~120Hz and the dynamic range of 60dB can be achieved by digitizing the EEG signal with a sampling rate of 250Hz and a quantization level of 11bit/sample, $1\mu V$/level. The compression ratio is defined as the ratio between the data volume after compression to that of the sampled and quantized waveform. The remaining problem for the method is to suppress the maximum error under $2\mu V_{\mathrm{p-p}}$.

The design of the method is based on the properties of EEG waveforms. Since an EEG is a multi-channel signal, each channel can be processed in parallel.

In this method, a one-channel EEG waveform is compressed by an original algorithm. That is to approximating the waveform by a mathematical function and store the coefficients of the function as compressed data. For efficient compression, appropriate approximation becomes an important problem.

The cycle of a waveform is not stable but varies in time. Thus, the first step in the method is to divide the waveform into segments, each of which has a similar cycle, so that the function approximation in the latter stage may succeed in much lower dimensinon. This division is done by evaluating the cycle of the low-passed waveform. The point where the waveform is divided into segments is called the "feature point."

The waveform can be considered as a combination of the basal wave component, which usually has a long cycle and large amplitude, and the superimposed wave component, which usually has a short cycle and small amplitude. Huge data volume was required to represent the small vibrations of the superimposed wave component precisely.

Thus the second step in the method is to decompose the waveform into the basal wave component and the superimposed wave component in order to reduce the bits required to quantize the superimposed wave component. This decomposition is done by approximating the waveform roughly. The rough approximation function becomes the basal wave component and the approximation error becomes the superimposed wave component.

In the next step, each of these decomposed waveforms are approximated by a mathematical function. The approximation function is a piecewise polynomial which can appropriately represent local variation. In this paper, fluency function [107] is used as an approximation function because it has less vibration on an approximated curve.

The error is evaluated by the least mean square because the approximation error of the result does not concentrate at one point, and produces a good approximating result in terms of computer complexity. The most efficient compression can be done by finding the approximation function with the smallest dimension which satisfies the tolerance error. The dimension is searched by a binary search while iterating the approximation.

2.3.2 Compression method

Figure 2.1 shows the outline of the compression method.

First, the axes are determined to digitize the original waveform. The sampling

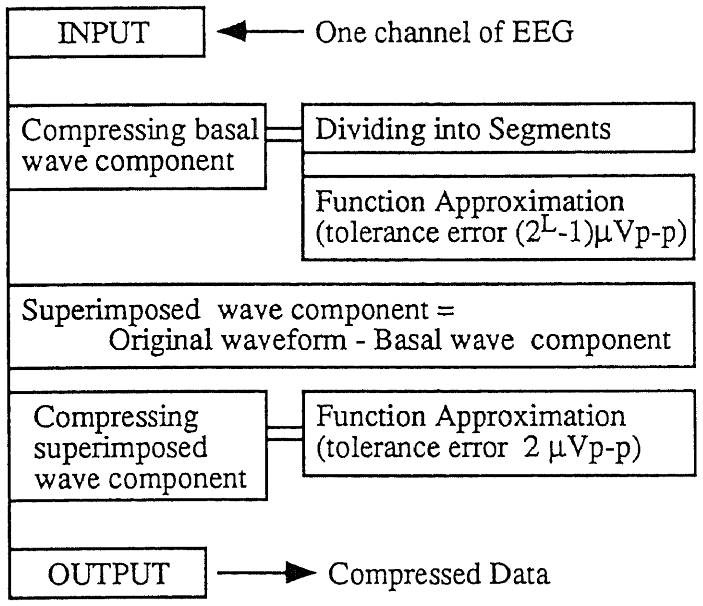

INPUT
One channel of EEG
Compressing basal wave component
Dividing into Segments
Function Approximation (tolerance error (2^L-1)μVp-p)
Superimposed wave component = Original waveform - Basal wave component
Compressing superimposed wave component
Function Approximation (tolerance error 2 μVp-p)
OUTPUT
Compressed Data

Figure 2.1: Compression procedure for EEG waveform.

points $\{t_i\}_{i=0}^{K-1}$ are set in equidistance.

$$t_i = i \times h, \qquad i = 0, 1, 2, \ldots, K-1$$

From the sampling rate of 250Hz, the sampling cycle h becomes h = 4ms. Let $z(t_i)$ be the sampled value of the waveform at each sampling point t_i. From the requirements for the dynamic range, the quantization bit length becomes 11 bits.

The waveform is divided into segments, each of which has a similar cycle. The division is done as follows. First, the original waveform is filtered by low-pass digital filter. By evaluating the cycle of low-passwd waveform, we can obtain the rough cycle of the waveform. The cut-off frequency of the low-pass filter is set to 18 Hz which is the typical frequency of a fast wave. Next, the first cycle is marked as the reference cycle. Let l_{ref} stand for this reference cycle. Each cycle is compared with the cycle l_{ref}, and while the cycle is in the range

$$[l_{\mathrm{ref}} - w, l_{\mathrm{ref}} + w],$$

it is considered to be in the same segment. If the cycle goes out of the range, the point is marked as a feature point, and the cycle becomes the next reference cycle. The tolerance variation w is determined as w = 60ms according to the experimental results. This segment is the basic unit which the compression and reconstruction processes are applied.

In the following, the process for one segment is described.

The EEG waveform is decomposed into a basal wave component and a superimposed wave component and approximated by a function. The decomposition and the approximation is processed as follows.

The decomposition is performed by roughly approximating the EEG waveform with a tolerance error of $(2^L - 1)\mu\mathrm{V}_{\mathrm{p-p}}$. The approximation function becomes the basal wave component, and the approximation error, which is the difference signal between the original EEG waveform and the basal wave component, becomes the superimposed wave component(Fig.2.2). Since the approximation error is limited

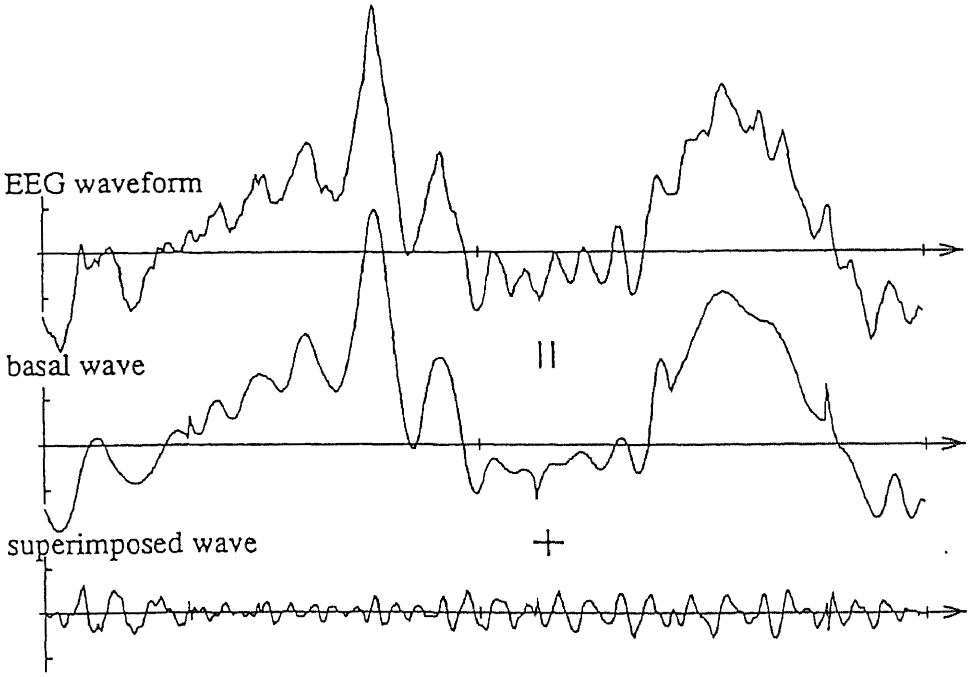


Figure 2.2: Dividing EEG waveform into two wave components.

under $(2^L - 1)\mu\mathrm{V}_{\mathrm{p-p}}$, the required quantization level to represent the superimposed wave component becomes Lbit. The constant L was determined as $L = 4$ according to the experimental results.

The superimposed wave component, which is obtained as an approximation error of the previous stage, is then approximated using fluency functions. This approximation is exactly identical to the method which was used in the previous step, except that the tolerance error, which determines the maximum error of the reconstructed waveform, is set to $2\mu\mathrm{V}_{\mathrm{p-p}}$.

In the following, the function approximation process is described in detail.

Let $\{v(t_i)\}_{i=0}^{N-1}$ stand for the signal being approximated.

The approximation function $s(t)$ for each segment between adjacent feature points is represented by the fluency function of degree 2.

Let $[0, T]$ be the segment between adjacent feature points. The approximation function $s(t)$ for the segment $[0, T]$ can be represented by a linear combination

$$s(t) = \sum_{k=-2}^{n-3} c_k \psi_k(t) \quad ,$$

where $\psi_k(t)$ is a local supported function:

$$\psi_k(t) = 3(T/n)^{-2} \sum_{q=0}^{3} \frac{(-1)^q}{q!(3-q)!}(t - (T/n)(k+q))_+^2, \\ k = -2, -1, \ldots, n-3 \quad .$$

Here, $(t-a)_+^2$ is a truncated power function defined as

$$(t-a)_+^2 \triangleq \begin{cases} (t-a)^2, & t > a \\ 0, & t \le a \end{cases} \quad .$$

The coefficients $\{c_k\}_{k=-2}^{n-3}$ of the approximation function $s(t)$ are determined by solving the regular equations

$$\sum_{k=-2}^{n-3} c_k \left\{ \sum_{i=0}^{N-1} \psi_k(t_i)\psi_l(t_i) \right\} = \sum_{i=0}^{N-1} v(t_i)\psi_k(t_i) \qquad l = -2, -1, \ldots, n-3.$$

The optimal degree n of the approximation function shall be the minimum degree which the error

$$|\max_{0\le i\le N-1}(s(t_i)-z(t_i))|+|\max_{0\le i\le N-1}(z(t_i)-s(t_i))|$$

becomes under the tolerance error. The optimal degree is searched by a binary search while iterating the approximation.

The length of segments, dimension n and the coefficients $\{c_k\}_{k=-2}^{n-3}$ are stored as compressed data.

The data for the basal waveform and for the superimposed waveforms are stored separately. To avoid distortion in the quantization, the effect of the rounding error of the coefficients to the function value shall be set at under 0.5.

The effect of the rounding error of coefficients c_k to the function value ([Property 1]) and the relation of the amplitude range of the approximation function $s(t)$ to that of the coefficients $\{c_k\}_{k=-2}^{n-3}$ ([Property 2]) are found in Ref.[108].

[Property 1]

Let c_k^* be the quantized representation of c_k in finite digits. The two functions $s(t)$ and $s^*(t)$ defined as

$$s(t)\triangleq\sum_{k=-2}^{n-3}c_k\psi_{k(t)}\quad ,$$

$$s^*(t)\triangleq\sum_{k=-2}^{n-3}c_k^*\psi_{k(t)}\quad ,$$

satisfy the following relations.

$$\max_{0\le t\le T}|s^*(t)-s(t)|\le\max_{-2\le k\le n-3}|c_k^*-c_k|$$

[Property 2]

The approximation function represented by

$$s(t)=\sum_{k=-2}^{n-3}c_k\psi_{k(t)}\quad ,$$

holds good that

$$\max_{-2\leq k\leq n-3} |c_k| \leq 2(\max_{0\leq t\leq T} |s(t)|) \quad .$$

From [Property 1], the stored data $\{c_k^*\}_{k=-2}^{n-3}$ is obtained by rounding $\{c_k\}_{k=-2}^{n-3}$ to an integer. Since the quantization level of the original waveform is 11 bits, the coefficients c_k^* for the basal wave component can be stored in 12 bits. The coefficients for the superimposed components can be stored in 5 bits.

2.3.3 Reconstruction method

The reconstructed waveform can be obtained by adding the basal wave component and the superimposed one for each segment. The reconstruction procedure is shown in Fig. 2.3. The basal wave component and superimposed one are reconstructed as follows.

Let $[0,T]$ be the segment for reconstruction. If the point t_i is in the domain $[(T/n)(M-1),(T/n)M]$, the value of the approximation function $s(t)$ at the point t_i is given as follows[109].

$$s(t_i) = \sum_{k=M}^{M+2} c_k^* \psi_k(t_i)$$

Here, $\psi_k(t_i)$ is represented by

$$\psi_k(t_i) = \begin{cases} 0.5 \times p^2 & k = M \\ p(1-p) + 0.5 & k = M+1 \\ p(0.5p-1) + 0.5 & k = M+2 \\ 0 & k \leq M-1, M+3 \leq k \end{cases}$$

$$p = M - t_i \times n/T.$$

2.4 Experimental Results

The performance of the proposed method was evaluated by applying it to nine typical clinical EEG waveforms selected at Toranomon Hospital.

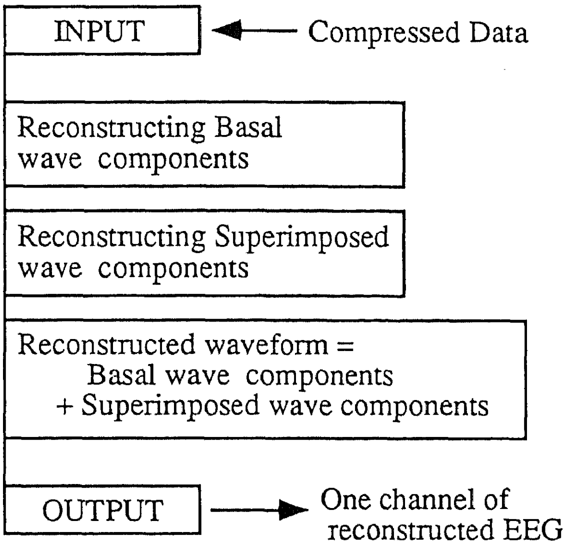

INPUT
Compressed Data
Reconstructing Basal wave components
Reconstructing Superimposed wave components
Reconstructed waveform = Basal wave components + Superimposed wave components
OUTPUT
One channel of reconstructed EEG

Figure 2.3: Reconstruction procedure for EEG waveform.

Table 2.2: Experimental results

waveform	comp. ratio	error (μV_{p-p})	processing time (sec) comp.	reconst.	gzip
(a) irregular slow wave	0.6128	2.0	6.060	0.113	0.928
(b) hump&spindle	0.4492	2.0	5.040	0.114	0.783
(c) large fast wave	0.6074	2.0	7.420	0.120	0.842
(d) spike	0.4858	2.0	3.050	0.126	0.450
(e) α-wave	0.4082	2.0	4.940	0.109	0.548
(f) fast wave & spike	0.6948	2.0	6.910	0.123	0.884
(g) small-amplitude β-wave	0.7183	2.0	3.610	0.097	0.658
(h) EEG with EMG	0.8535	2.0	11.140	0.120	0.974
(i) EEG with strong EMG	1.0000	0.0	21.200	0.053	0.939
average	0.6478	1.78	7.708	0.108	0.778

The waveforms were digitalized by a sampling rate of 250Hz and a quantization level of 16bit/sample, $1\mu V$/level and temporaliry stored in a harddisk. The compression and reconstruction methods were implemented on a workstation Sun4/1 (12.5MIPS, SunOS4.0.3), and the compressed data were stored in an optical disk.

Figure 2.4 shows the reconstructed waveforms with the original ones. Table 2.2 shows the experimental results for the quality, compression ratio, and the processing time. The compression ratio by conventional compression method (gzip1.2.4) is also shown for comparation. The proposed method achieves better compression ratio on the average.

The bandwidth of 0.5 ~ 120Hz and the dynamic range of 60dB were achieved by a sampling ratio of 250Hz and the quantization level of 11bit/sample. The compression ratio varies for the complexity of the waveform, and the average was 0.648. The processing speed for 1024 points, or about 4 seconds, is 7.708 seconds

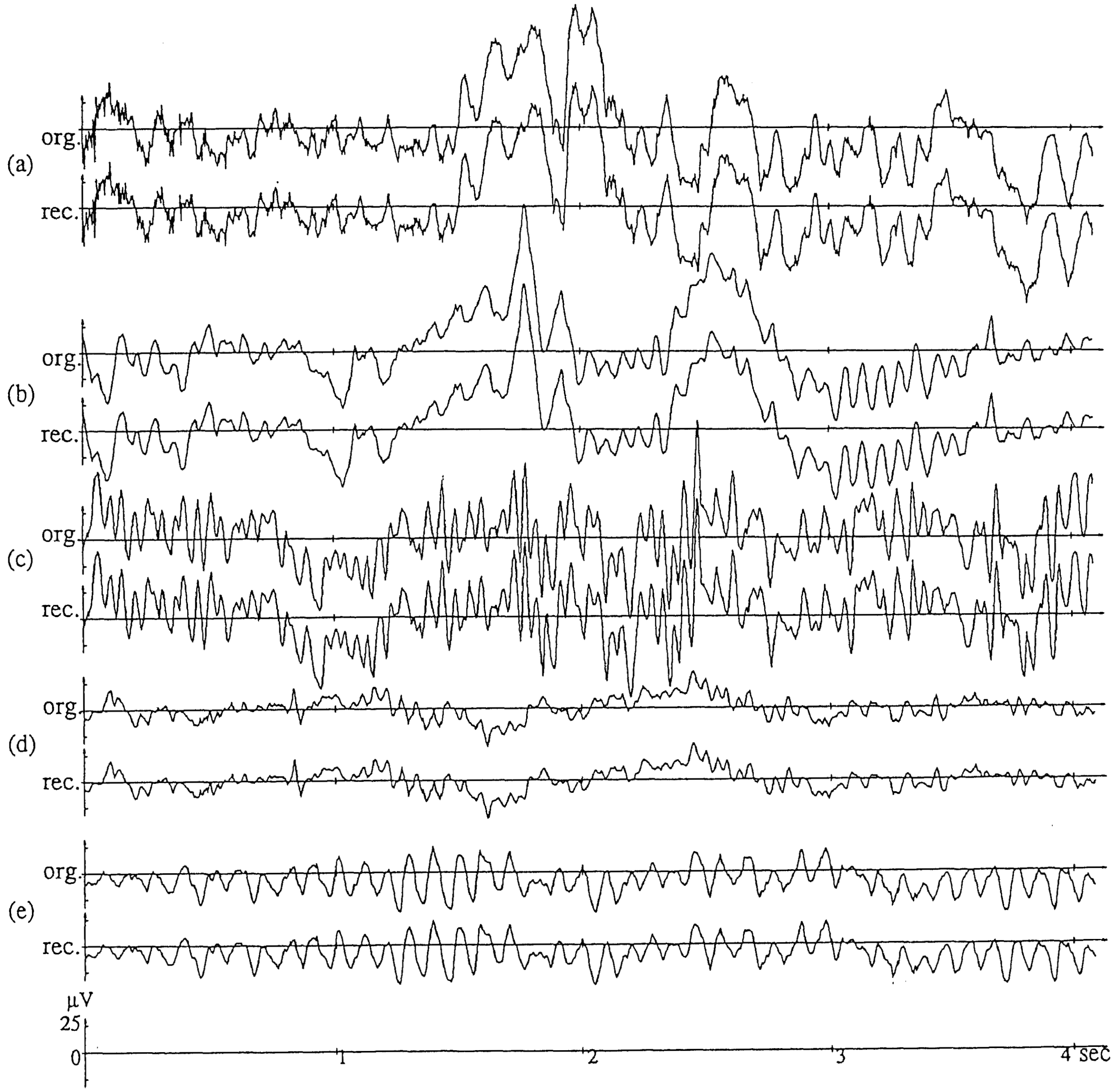


Figure 2.4: Original and reconstructed waveforms

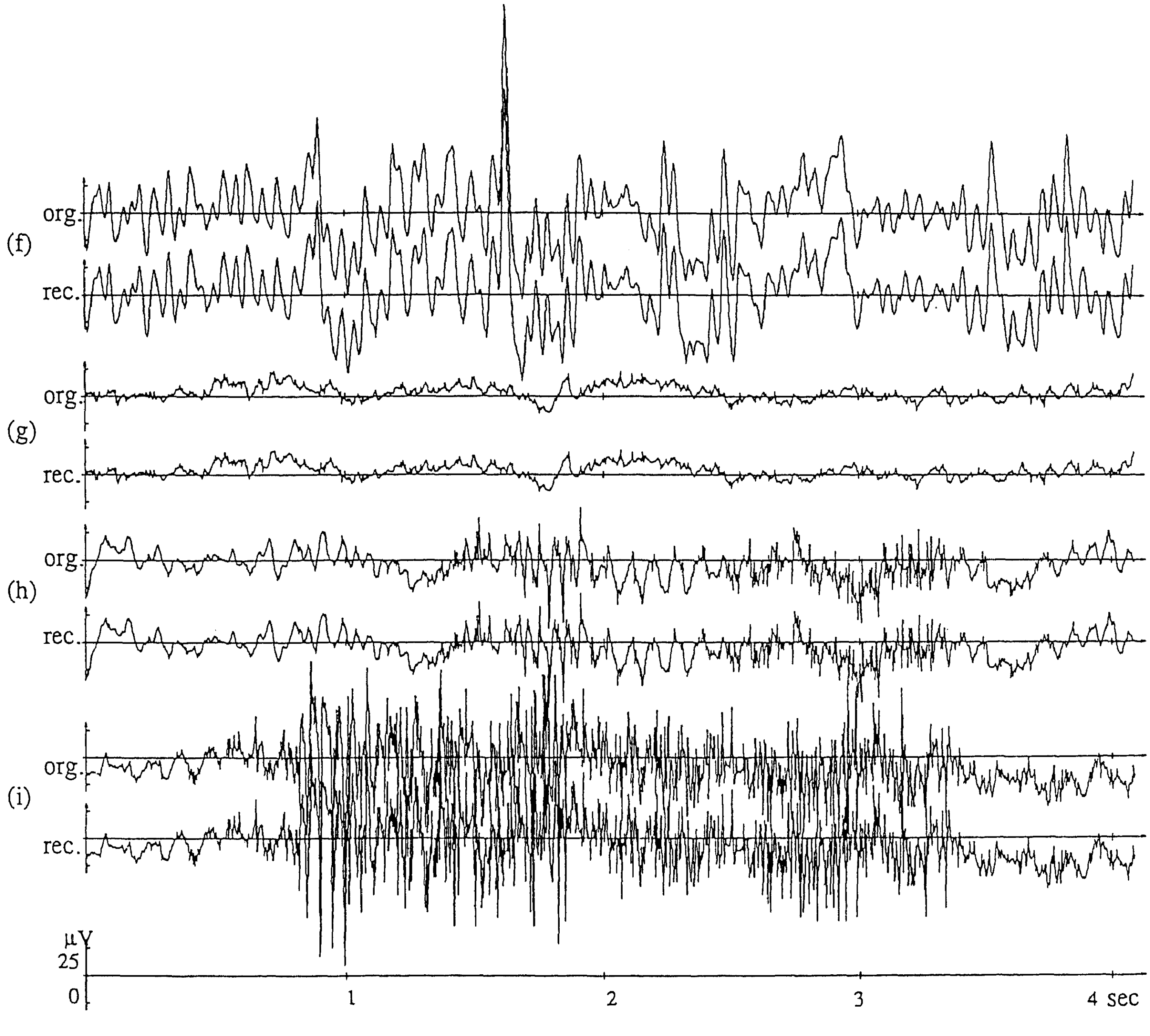


Figure 2.4: Original and reconstructed waveforms(continue)

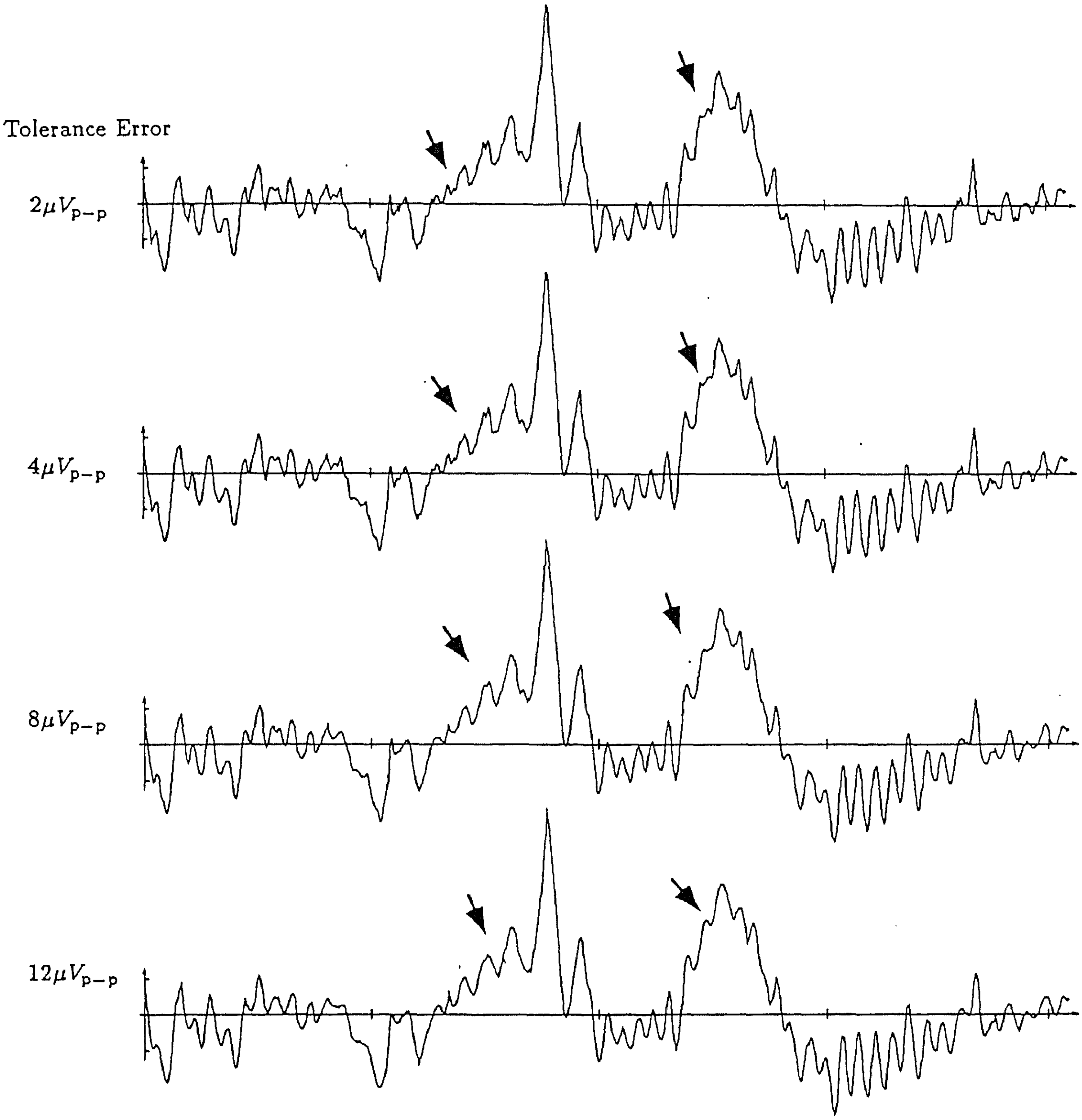


Figure 2.5: Reconstructed waveforms under different tolerance error.

Table 2.3: Relation between the tolerance error and the compression ratio

Tolerance Error ($\mu V_{\mathrm{p-p}}$)	2	4	8	12
Compression Ratio for waveform (a)	0.4492	0.4019	0.3047	0.2637
Average for nine waveforms	0.6478	0.6053	0.5471	0.4887

for compression and 0.108 seconds for reconstruction.

Table 2.3 shows the relation between tolerance error and the compression ratio. The reconstructed waveform is shown in Fig. 2.5. The arrow shows the point where the quality is lost.

The experimental result shows that the achieved quality is as high as that for a conventional recording device. Clinicians also say that these reconstructed waveforms have sufficient quality for diagnosis. The method therefore satisfies the specification determined in Section 2.2.

2.5 Summary

A typical recording for a patient can be stored in 7M bytes, which was 11M bytes before compression. This makes it possible to store EEG data for 110 patients on an 800 megabyte optical disk, which could only store data for 70 patients.

The processing speed means that reconstruction is fast enough to reconstruct in real time, while compression cannot be done so quickly. Real time reconstruction is possible if this algorithm is modified to process in parallel or implemented in hardware. Most of the time taken for compression is spent in determining the optimal dimension.

Further research is required for the dynamic setting of appropriate tolerance error, enhancing the speed of the determining process for the optimal dimension, improvement of hardware implementable algorithm, and for the utilization of correlation between each channel.

Compression Methods for Two-Variable Signal

Chapter 3

An Automatic Function-Font Generating System

In this chapter, a compression method for binary image has been proposed as an application for the compression method for a two-variable signal.

3.1 Introduction

The written word is the next oldest means of communication to the spoken word[113]. Since Gutenberg invented the printing machine in the 15th century, the notions of "type" and "typesetting" have developed and radically changed the efficiency of printing. Typesetting has changed from hand-composition to phototypesetting. Although some methods of typesetting are even computer controlled, a phototypesetter is basically an optical system consisting of mirrors, lenses and so on. Since the image of a character is processed as an analog signal, the output image has very high resolution. However, enlarging a character using optical lenses has a physical limitation: it cannot enlarge the image without distortion.

Recently, computer aided electronic digital printing has become feasible with the increasing power of the computer[111]. There are two kinds of font which a computer can handle: bitmap fonts and scalable fonts. A bitmap font is a picture of the font which has been optimized to look good at a specific size. Though it is a compact way of representing fonts when they are small in size, it cannot be scaled to other sizes. Scalable fonts, on the other hand, are defined mathematically. There are no limitations in enlarging the character because it can be rendered at any requested size. In creating high quality documents, many fonts with various typefaces and sizes are required. Considering the required storage, it is obvious that we should use scalable fonts. However, the creation of scalable fonts is not an easy task. Even an expert takes about an hour to create one typeface for a single character. In Japan, since there are more than 7,000 characters defined in JIS (Japanese Industrial Standard) character set, it takes about a year to develop a scalable font for one typeface.

The ultimate goal of this research is to develop a system which automatically generates scalable fonts from a given bitmap image. Generating a scalable font from a bitmap image can be thought of as extracting the essence of the typeface from the value at sampled points.

Preliminary research, prior to this study, involved an experiment on automatic function-font generating system for brush-written characters and for printed character and figures[5]. Through these investigation,s several problems in constructing a practical system became clear. In this chapter, a practical system to generate function-fonts for printed characters and figures is proposed.

3.2 Specifications

Figure 3.1 show the outline of the initial experiment system. First, the boundary points are tracked from the bitmap image. Several sets of boundary points are obtained. Each set of boundary points is approximated by fluency-functions. To keep the quality of the bitmap image, some feature points are extracted in advance of function approximation. These points are called "joint points" because they are places where two functions are joined. Finally, the coefficients of the approximation function become output.

The initial experiments indicated the feasibility of a practical system. However, the following problems also became clear.

- In the initial experiments, the size of bitmap was fixed to 256×256 pixels. We found that this size is not sufficient to generate function-fonts for complicated characters.

- In the initial experiments, the bitmap images were treated as noiseless images. Usually, however, the bitmap image read by a scanner contains several kinds of noise. Therefore, an automatic or interactive noise reduction method is required.

- In the initial experiments, the boundary points were simply tracked using the eight-neighboring connection rule. This caused a serious problem in the approximating stage when the image becomes more complicated. An improved tracking algorithm should be developed.

- In the initial experiments, the location of joint points were always snapped to the location of a boundary point. Since the boundary points are just a sampling point, the location of a joint point should be free from them.

- In the initial experiments, the tangent lines of the approximate function at joint points are not considered at all. This causes an insufficiently smooth connection and produced an unsatisfactory effect when the function-fonts were reconstructed in large sizes.

The design of the actual system proposed takes the above into account, as shown in Fig. 3.2. In the next section, each module is described in detail.

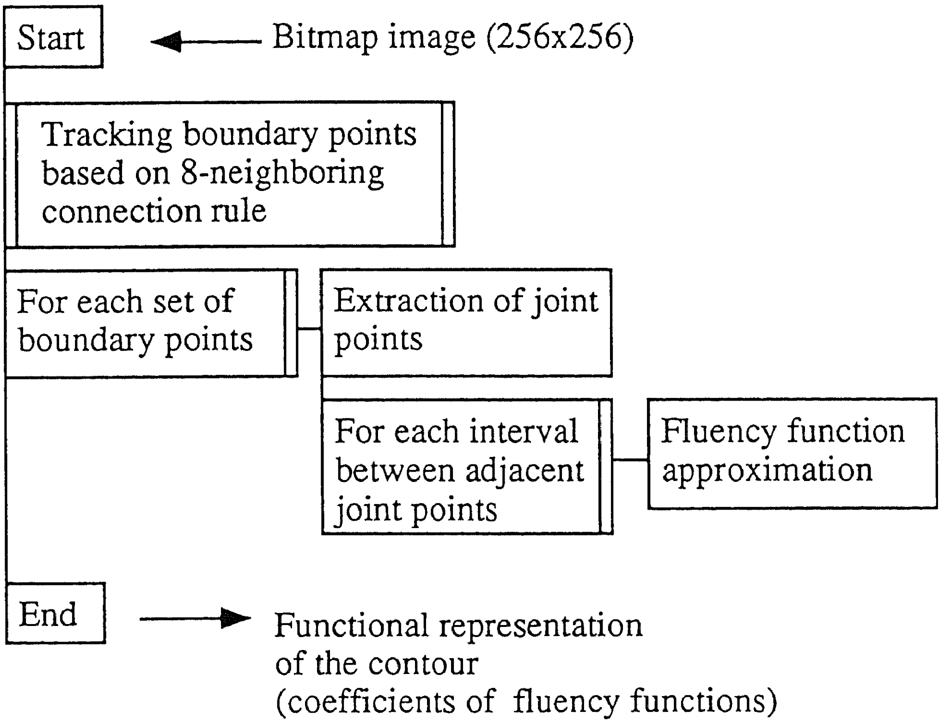

Start
Bitmap image (256x256)
Tracking boundary points based on 8-neighboring connection rule
For each set of boundary points
Extraction of joint points
For each interval between adjacent joint points
Fluency function approximation
End
Functional representation of the contour (coefficients of fluency functions)

Figure 3.1: Structure of the initial experiment system

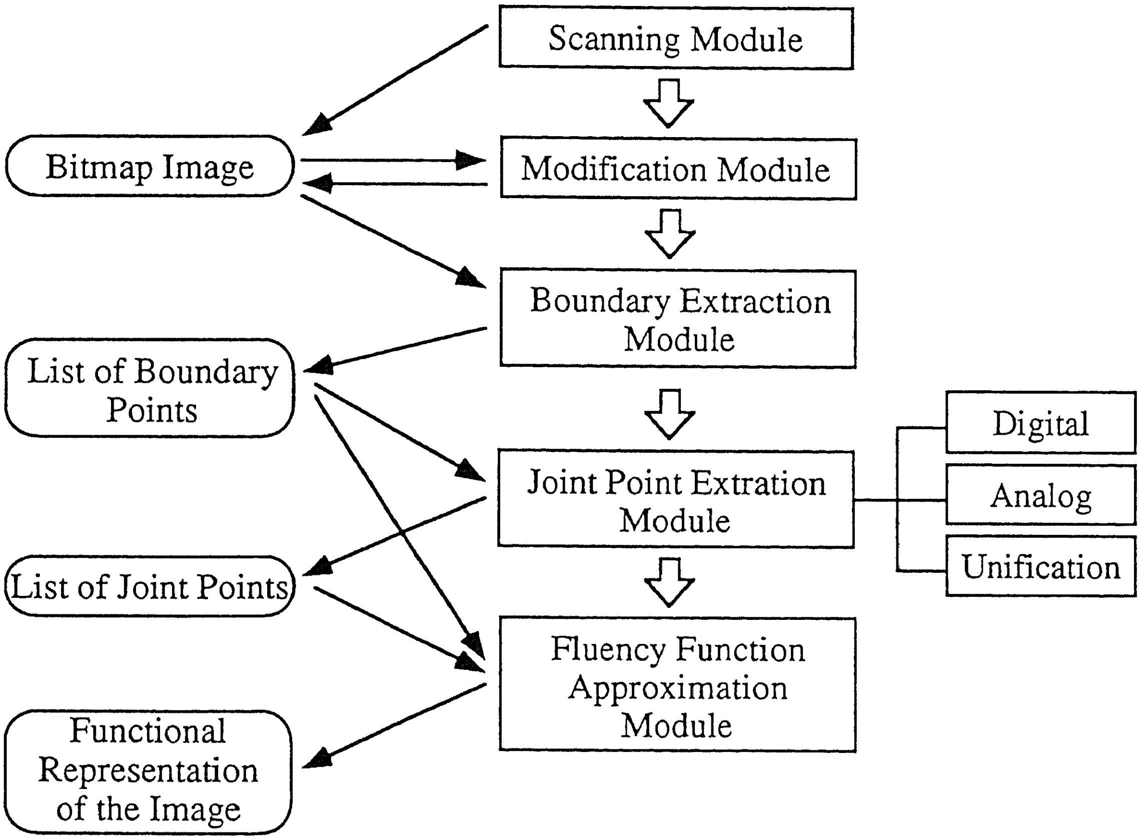

Scanning Module
Bitmap Image
Modification Module
Boundary Extraction Module
List of Boundary Points
Joint Point Extration Module
Digital
Analog
Unification
List of Joint Points
Fluency Function Approximation Module
Functional Representation of the Image

Figure 3.2: Structure of the function-font generating system

3.3 Compression and Reconstruction Methods

3.3.1 Compression method

3.3.1.1 Scanning module

Most font designers usually design their fonts on a sheet of paper. Therefore the original image should be read by an optical reader. The scanning module reads the image of a character using an optical reader and writes a bitmap file as an output.

It is obvious that the larger the size of a bitmap becomes, the more accurate approximation is available. The size of a bitmap is controlled by the size of the original image and by the scanning resolution. The size of the original image cannot change because it depends on the designer's skill. If the size is small, the design cannob be precise. But if it is too large, the balance of the typeface cannot be maintained. The resolution should therefore be controllable.

3.3.1.2 Bitmap modification module

In the previous work in the Wisdom Systems Laboratory, the input images were treated as noiseless. However, a bitmap image which is scanned by an optical reader contains several kinds of noise. As is expected, the higher the resolution becomes, the worse the obtained bitmap image becomes. This is because the sensitivity to noise becomes higher. Therefore, a noise reduction method should be provided. The modification module provides noise reduction features for the following noises.

Ragged edges: The edges of a scanned bitmap image are usually jaggy(Fig. 3.3). This kind of noise can be eliminated by a template matching method.

Isolated dots: Isolated dots are often generated. The most established methods to eliminate such noise are dilatation and shrinking. It can also be eliminated by the template matching method.

Rotation: As the resolution becomes high, it becomes difficult to adjust the sheet vertically or horizontally. The module provides a function to slightly rotate the image so as to straighten the marked points vertically or horizontally.

The module also provides a bitmap editor with other primitive operations such as shift, flip, inverse and so on.

3.3.1.3 Boundary tracking module

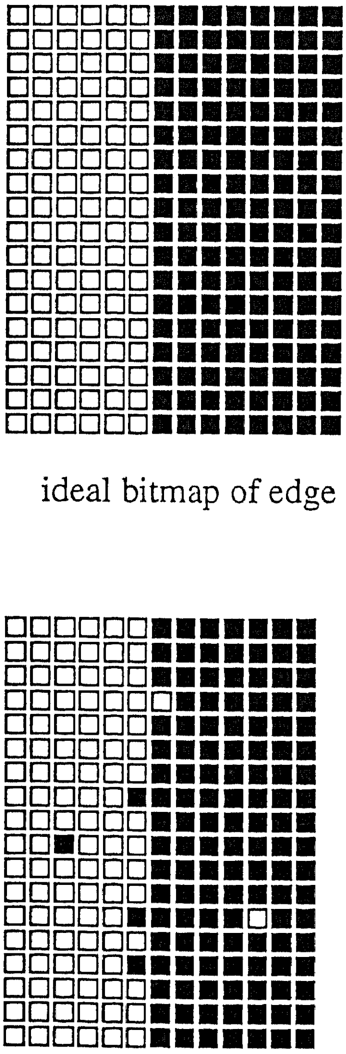
This module reads a bitmap image as an input and writes sets of boundary points as an output.

In the previous work in our laboratory, the boundary points were tracked based on eight-neighboring tracking. This method has disadvantages. One is that the black pixel and the white pixels are not treated symmetrically. Even if the shape of a region is the same, the shape of the boundary results differ. Another problem occurs in cases where the width of the black area is only one pixel. Tracking such an image may generate overlapping contours, or a region with no width. These are shown in Fig. 3.4(a).

To avoid these problems, the boundary points are tracked by using 2×2 masks [116], shown in Fig. 3.4(b), which is based on an edge following method. Though this does not give complete symmetry, it does give better results (Fig. 3.4(c)) with slight increase of data.

3.3.1.4 Joint point extraction module

The joint point extraction module reads a set of boundary points as an input and writes a set of joint points as output. For each interval between adjacent joint points, a label, which indicates which kind of function would be suitable for approximation, is also given as output. In the following, an algorithm to extract appropriate joint points is described in detail.



ideal bitmap of edge

bitmap of edge
actually obtained

Figure 3.3: Noises in bitmap images

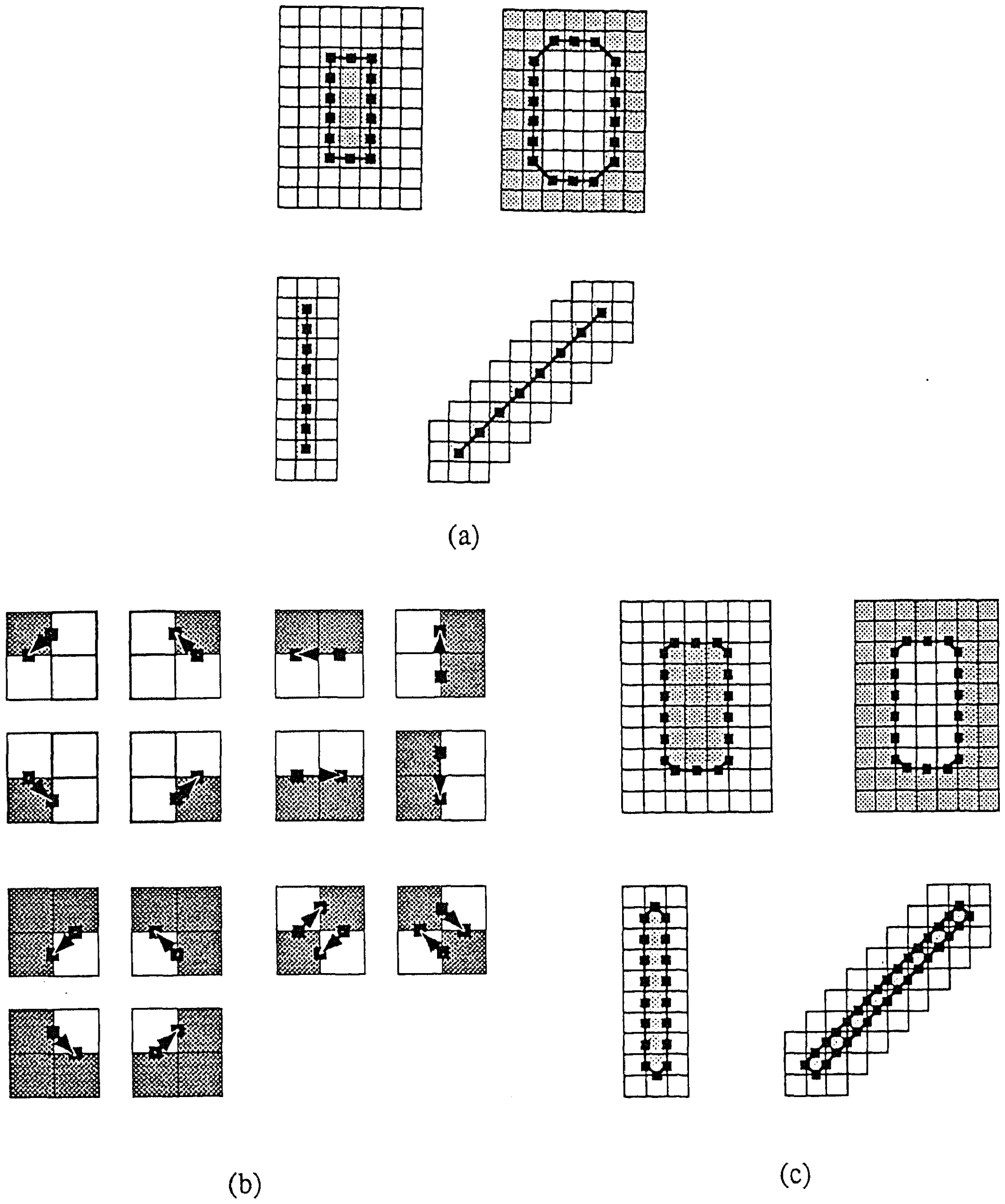


Figure 3.4: Boundary point extraction

(a)boundary points extracted by eight-neighboring connection rule.

(b)2×2 masks for edge-following.

(c)boundary points extracted by the present method.

Figure 3.5 shows the outline of the extraction module. The module consists of 3 stages. In the first stage, obvious joint point candidates are extracted at right angled corners and at both ends of a straight line, by evaluating the digital curvature. In the second stage, additional joint point candidates are extracted where the kind of approximation function, such as a straight line, arc, or other curved line, should be changed. In the third stage, redundant joint point candidates are removed.

The algorithm differs from that of initial experiments in the following two ways.

- In this method, the point marked as a joint point candidate only means that the final joint point exists near the point. In other words, a point with no mark may become a final joint point instead of the marked one.
- The final joint points are not restricted to the boundary points. This makes it possible to find a more suitable joint point location, which minimizes the approximation error in the approximation module.

Extracting Joint Points Based on Digital Curvature Let $\{(x_{i_1}, y_{i_1})\}_{i_1=1}^{n_1}$ denotes a set of boundary points.

First, the right angled corners are extracted by evaluating the digital curvature at every point. The digital curvature is calculated as

$$P_{i_1K_1} = \frac{a_{i_1K_1} \cdot b_{i_1K_1}}{|a_{i_1K_1}|\,|b_{i_1K_1}|}, \qquad i_1 = 1, 2, ..., n_1$$

where

$$a_{i_1K_1} = (x_{i_1+K_1} - x_{i_1}, y_{i_1+K_1} - y_{i_1}),$$

$$b_{i_1K_1} = (x_{i_1-K_1} - x_{i_1}, y_{i_1-K_1} - y_{i_1}).$$

Here the symbol '$\cdot$' denotes the inner product of vectors. The point which satisfies the condition of $P_{i_1K_1} = 0$ is considered as a right angled corner and marked as a joint point candidate. The constant K_1 is set as $K_1 = 3$ so that the candidates

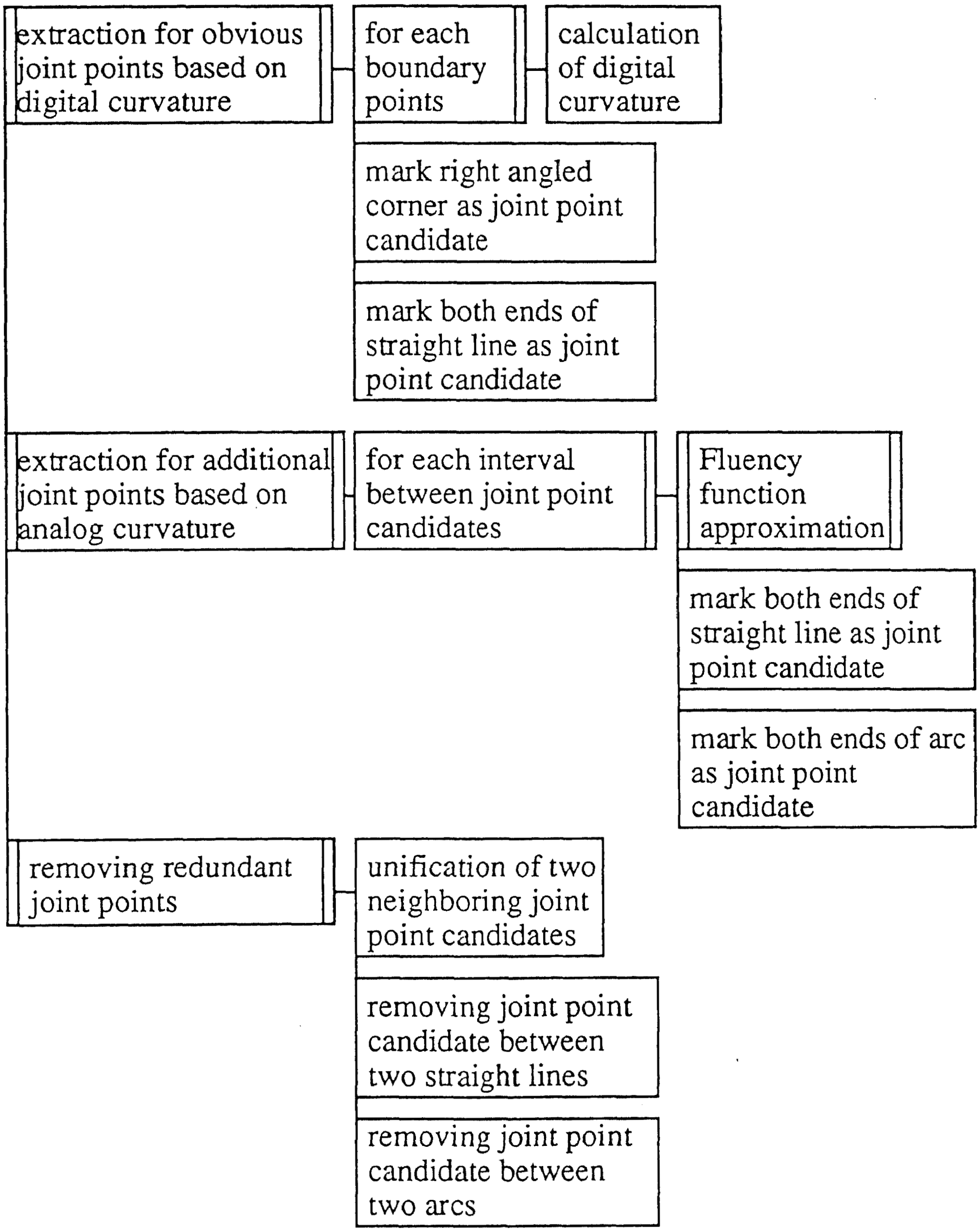

extraction for obvious joint points based on digital curvature
for each boundary points
calculation of digital curvature
mark right angled corner as joint point candidate
mark both ends of straight line as joint point candidate
extraction for additional joint points based on analog curvature
for each interval between joint point candidates
Fluency function approximation
mark both ends of straight line as joint point candidate
mark both ends of arc as joint point candidate
removing redundant joint points
unification of two neighboring joint point candidates
removing joint point candidate between two straight lines
removing joint point candidate between two arcs

Figure 3.5: Multi-stage joint point extraction procedure

are extracted at the appropriate location where human would also think the joint point must be there.

Next, both ends of straight lines are extracted as joint point candidates. If a point which satisfies the condition of $P_{i_1 K_1} = -1$ continues for more than K_2 points, the sequence of $K_2 + 2 \times K_1$ points is considered as an obvious straight line. Both ends of this sequence are marked as joint point candidates. The sequence is also labeled as straight line candidate. The constant K_2 is set as $K_2 = 25$. The joint point candidates are extracted as $\{(x_{i_2}^{(J)}, y_{i_2}^{(J)})\}_{i_2=1}^{n_2}$. Here n_2 is the number of joint points extracted in this stage.

Extracting joint points based on analog curvature In the second stage, additional joint points are extracted at the points of connection of different kinds of lines. Since the connection may be smooth, the point cannot be decided by the digital curvature. The analog curvature is therefore temporarily calculated at each boundary point. The second stage consists of two processes. First, a set of boundary points between each pair of adjacent joint points is approximated by fluency function of degree 2. Then, by evaluating the curvature of the function approximating the boundary points, the connections between different classes of the line are decided and marked as additional joint points. In the following, each process is described in detail.

Function approximation: Let $\{(x_{i_3}, y_{i_3})\}_{i_3=1}^{n_3}$ stand for a set of boundary points between each pair of adjacent joint points. Here n_3 means the number of points in the sequence. One convenient method to represent a curve on 2-dimensional space is parametric expression. Here we are going to approximate the sequence with $(s_x(t),\ s_y(t))$ with parameter t, where $s_x(t)$ and $s_y(t)$ are fluency functions of degree 2 approximating the points

$$\{(t_{i_3}, x_{i_3})\}_{i_3=1}^{n_3}, \qquad t_{i_3} = i_3 - 1$$

and

$$\{(t_{i_3}, y_{i_3})\}_{i_3=1}^{n_3}, \qquad t_{i_3} = i_3 - 1$$

respectively.

The fluency function of degree 2 is identical with a spline function of degree 2. It is a piecewise polynomial and is continuously differentiable in the observation domain [0,T]. The joints of the pieces are called knots, $\{\xi_k\}_{k=-2}^{n}$, and they are given by $\xi_k = (T/n)k$ $(k = -2, -1, \ldots, n)$, where n means the dimension of s. The function s can be represented by

$$s(t) = \sum_{k=-2}^{n-3} c_k N_{k,3}(t)$$

where $N_{k,3}(t)$ is a normalized B-spline function defined as follows:

$$N_{k,3}(t) \equiv 3(T/n) - 2 \sum_{p=0}^{3} (-1)^p \frac{(t - \xi_k + p)_+^2}{p!(3-p)!}.$$

Here, $(t-a)_+^2$ is a truncated power function defined as

$$(t-a)_+^2 \triangleq \begin{cases} (t-a)^2, & t > a \\ 0, & t \leq a \end{cases} .$$

Coefficients $\{c_k\}_{k=-2}^{n-3}$ are called B-spline coefficients.

When $n \geq 3$, the parameters $\{c_{x_{i_4}}\}_{i_4=-2}^{n-3}$ and $\{c_{y_{i_4}}\}_{i_4=-2}^{n-3}$ of $s_x(t)$ and $s_y(t)$ are determined to minimize the square error Q:

$$Q = \sum_{i_3=1}^{n_3} |x_{i_3} - s_x(t_{i_3})|^2 + \sum_{i_3=1}^{n_3} |y_{i_3} - s_y(t_{i_3})|^2$$

as

$$\sum_{k=-2}^{n-3} c_{x_k} \{\sum_{i_3=1}^{n_3} N_{k,3}(t_{i_3}) N_{l,3}(t_{i_3})\} = \sum_{i_3=1}^{n_3} x_{i_3} N_{l,3}(t_{i_3}), \quad l = -2, -1, \ldots, n-3$$

$$\sum_{k=-2}^{n-3} c_{y_k} \{\sum_{i_3=1}^{n_3} N_{k,3}(t_{i_3}) N_{l,3}(t_{i_3})\} = \sum_{i_3=1}^{n_3} y_{i_3} N_{l,3}(t_{i_3}), \quad l = -2, -1, \ldots, n-3$$

respectively.

In order to judge the convergence of the least mean squares approximation, the dimension n is increased until

$$\varepsilon = \max_{0\le t_{i_3}\le n_3-1} \sqrt{(s_x(t_{i_3}) - x_{i_3})^2 + (s_y(t_{i_3}) - y_{i_3})^2}$$

becomes $\varepsilon < 0.90$.

Deciding joint points: In this process, joint points at connections between different kinds of line are extracted by finding both ends of straight lines and of arcs.

The next property shows equations to find the curvature analytically from the coefficients of the approximate functions.

(Property) Let the approximation function represented in the form

$$s_x(t) = \sum_{k=-2}^{n-3} c_{x_k} N_{k,3}(t).$$

Here, $\xi_k = (T/n)k$ is given as equispaced knots in the observation domain $[0, T]$. Let $s_x(t)$ is also represented in the form

$$s_x(t) = \alpha_x t^2 + \beta_x t + \gamma_x.$$

The $\{\alpha_x, \beta_x, \gamma_x\}$ are determined as

$$\begin{aligned}
\alpha_x &= \left.\frac{d^2 s_x(t)}{dt^2}\right|_{t=t_0} /2, \\
\beta_x &= \left.\frac{d^2 s_x(t)}{dt^2}\right|_{t=t_0} - 2\alpha_x t_0, \\
\gamma_x &= s_x(t_0) - (\alpha_x t_0 - \beta_x) t_0,
\end{aligned}$$

$$t_0 \in [\xi_k, \xi_{k+1}]$$

Then the j-th derivation of the function of degree $(m-1)$ in domain $[\xi_0 \leq t \leq \xi_{n-2}]$ is determined as

$$\frac{d^j S_x(t)}{dt}\left\{\sum_{k=-2}^{n-3} c_{x_k} N_{k,m}(t)\right\} = \sum_{k=j+1-m}^{n-3} c_{x_k}^{(j)} N_{k,3}(t),$$

where

$$c_{x_k}^{(j)} = \begin{cases} c_{x_k} & (j=0) \\ \frac{(m-j)(c_{x_k}^{(j-1)} - c_{x_{k-1}}^{(j-1)})}{\xi_{k+m-j+1} - \xi_k} & (j = 1, 2, \ldots, m-1) \end{cases}$$

From this property, the curved line between joint points are represented in parametric expressions. When $s_x(t)$ and $s_y(t)$ are represented as

$$s_x(t) = \alpha_x t^2 + \beta_x t + \gamma_x,$$

$$s_y(t) = \alpha_y t^2 + \beta_y t + \gamma_y,$$

the curvature $\kappa(t), t \in [\xi_k, \xi_{k+1}]$ is determined as

$$\begin{aligned} \kappa(t) &= \frac{s_x'(t)s_y''(t) - s_x''(t)s_y'(t)}{\{s_x'(t)^2 + s_y'(t)^2\}^{3/2}} \\ &= \frac{2(\beta_x \alpha_y - \alpha_x \beta_y)}{\{(2\alpha_x t + \beta_x)^2 + (2\alpha_y t + \beta_y)^2\}^{3/2}}. \end{aligned}$$

The additional joint points are extracted by using this curvature as follows.

First, both ends of straight lines are extracted as joint points. The curvature $|\kappa(t_{i_3})|$ is calculated for each pair of points. If a point which satisfies the condition of $|\kappa(t_{i_3})| < K_3$ continues for more than K_4 points, that sequence is considered as a straight line. Let $[t_{s_1}, t_{e_1}]$ be this interval, and let $t_{s_1}^*$ and $t_{e_1}^*$ be both ends of this interval which are rounded off to the decimal point. Then the beginning point $(x_{t_{s_1}^*} + 1, y_{t_{s_1}^*} + 1)$ and the ending point $(x_{t_{e_1}^*} + 1, y_{t_{e_1}^*} + 1)$ are also extracted as joint points. The sequence is labeled as candidate for straight line. The constants K_3 and K_4 are set as K_3=1/200 and K_4=30.

Next, both ends of an arc are extracted as joint point candidates. The curvature $|\kappa(t_{i_3})|$ is evaluated for each pair of points. Let $\tilde{\kappa}$ be the mean value of evaluated curvature in the interval so far. The value $\tilde{\kappa}$ is renewed every time the curvature is evaluated, until κ is out of the range $[\tilde{\kappa} - K_5, \tilde{\kappa} + K_5]$. The sequence of points which has been evaluated so far is estimated as a candidate for an arc. Based on the mean value κ, the central angle of the arc which should fit the sequence is calculated. If the central angle is greater than K_6, the interval is considered as an arc. Both ends of the sequence are marked as joint points, and the sequence is labeled as a candidate for an arc. The constants are set as $K_5 = 3/400$ and $K_6 = \pi/2$ so that short straight lines or free curved lines are not estimated as arcs.

After applying these processes to all boundary points in the interval, the joint points at ends of straight lines and of arcs are extracted.

After all of the intervals are processed, the additional joint points are extracted as $\{(x_{i_2}^{(J)}, y_{i_2}^{(J)})\}_{i_2=n_2+1}^{n_4}$. Here n_4 is the total number of joint points extracted in two stages.

Removing redundant joint points In the previous two stages, many joint point candidates are extracted. As mentioned before, these candidates only indicate that the real joint points exist somewhere close to them. In this stage, redundant joint point candidates are removed by unifying candidates in close proximity to each other, or by replacing continuous lines(arcs) with one line(arc). This makes it possible to reduce the size of stored data.

Unifying close joint point candidates If more than two joint point candidates exist close to each other, it is very likely that they are actually a single joint point. If two joint point candidates exist within a distance of K_7 they are considered to be unified.

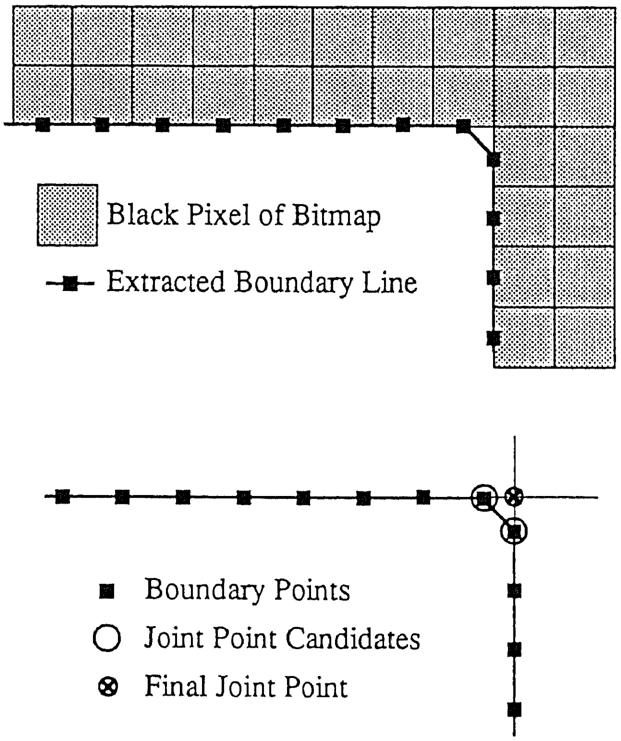


Figure 3.6: Unifying two joint points

Let two joint point candidates $P_i^{(B)}, P_j^{(B)}, (i < j)$ exist within a distance of K_7. The tangent lines for the outer side of each joint point are decided by calculating the line passing through points $P_i^{(B)}, P_{i-K_8}^{(B)}$ and points $P_j^{(B)}, P_{j+K_8}^{(B)}$, respectively. The intersection of the two tangent lines becomes a joint point candidate (Fig. 3.6).

Unifying continuous straight lines Redundant joint points between continuous straight lines are removed if the boundary points can be fitted with one straight line.

Let $\{(x_{i_4}^{(B)}, y_{i_4}^{(B)})\}_{i_4=ns}^{ne}$ be the joint points involved in continuing two or more candidates for a straight line.

Connect the two joint points $(x_{ns}^{(B)}, y_{ns}^{(B)})$ and $(x_{ns+2}^{(B)}, y_{ns+2}^{(B)})$. If the distance between the joint point $(x_{ns+1}^{(B)}, y_{ns+1}^{(B)})$ and the line is less than K_9, the joint point $(x_{ns+1}^{(B)}, y_{ns+1}^{(B)})$ is marked as a redundant joint point (Fig. 3.7(a)). If more than three straight lines continue, the distance of the joint point is evaluated by the greatest distance. The constant K_9 is set as $K_9 = 2$.

Unifying continuous arcs Redundant joint points between continuous arcs are removed if the boundary points can be fitted with one arc or a circle.

If two arc candidates with the same direction (clockwise or not) are continuing and if the distance between the centers of two arcs is less than K_{10} and the difference of radius is less than K_{11}, the two arcs can be merged into one arc. This is shown in Fig.3.7(b). The constants K_{10} and K_{11} are set as $K_{10} = 1$ and $K_{11} = 2$.

3.3.1.5 Fluency function approximation module

In the fluency function approximation module, each interval between adjacent joint point candidates is approximated.

This module reads each set of boundary points, list of joint point candidates and labels for each interval between adjacent joint point candidates. All intervals between adjacent joint points will have a label which indicates what kind of line

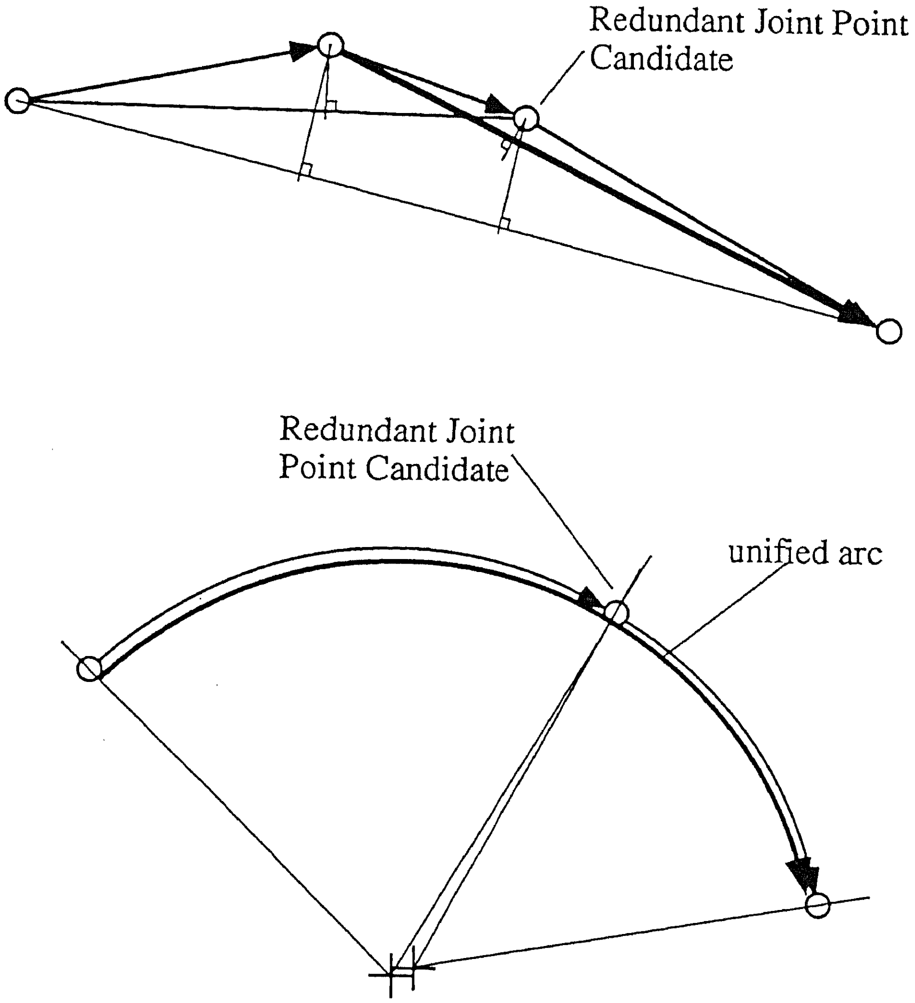

Redundant Joint Point Candidate
Redundant Joint Point Candidate
unified arc

Figure 3.7: Reduction of redundant joint points

(a) Removing redundant joint points between two straight lines

(b) Removing redundant joint points between two arcs should be used to approximate it, i.e., a straight line or an arc. If an interval has no label it means that the kind of line has not been determined.

At first, each interval which is labeled as a straight line candidate is approximated. The coordinate of the joint points at both ends of the line will become the stored data.

Next, each interval which is labeled as an arc is approximated.

The lack of smoothness at the ends of arcs loses the quality of the reconstructed result markably. If the neighboring interval is labeled as a straight line and the difference of inclination of the tangent line for both sides of the joint points is less than K_{12}, the joint point is relocated so that the line and arc connect smoothly (Fig.3.8).

The center point and radius of the arc are determined so as to make the mother circle touch the line and to minimize the approximation error for the arc. The constant K_{12} is set as $K_{12} = 0.2$. The coordinate of the center point, start point and end point and the direction of arc (clockwise or not) becomes the stored data.

Finally, each interval which has no label is approximated. The interval is first approximated by a straight line. If the approximation error

$$\varepsilon = \max_{0 \leq t_{i_3} \leq n_3 - 1} (s_x(t_{i_3}) - x_{i_3})^2 + (s_y(t_{i_3}) - y_{i_3})^2$$

is $\varepsilon < 0.90$, the interval is approximated by a line. If the neighboring interval is also approximated with a line, the joint point is renewed to be at the intersection of the both lines in order to keep the boundary continuous.

Otherwise, the interval is next approximated with an arc. The approximation error

$$\varepsilon = \max_{0 \leq t_{i_3} \leq n_3 - 1} (s_x(t_{i_3}) - x_{i_3})^2 + (s_y(t_{i_3}) - y_{i_3})^2$$

is evaluated to judge the converge. The center angle of the arc is changed from 1 to 1/6 until ε becomes $\varepsilon < 0.90$. The range from 1 to 1/6 comes from so as not to evaluate short straight line or curved line as an arc.

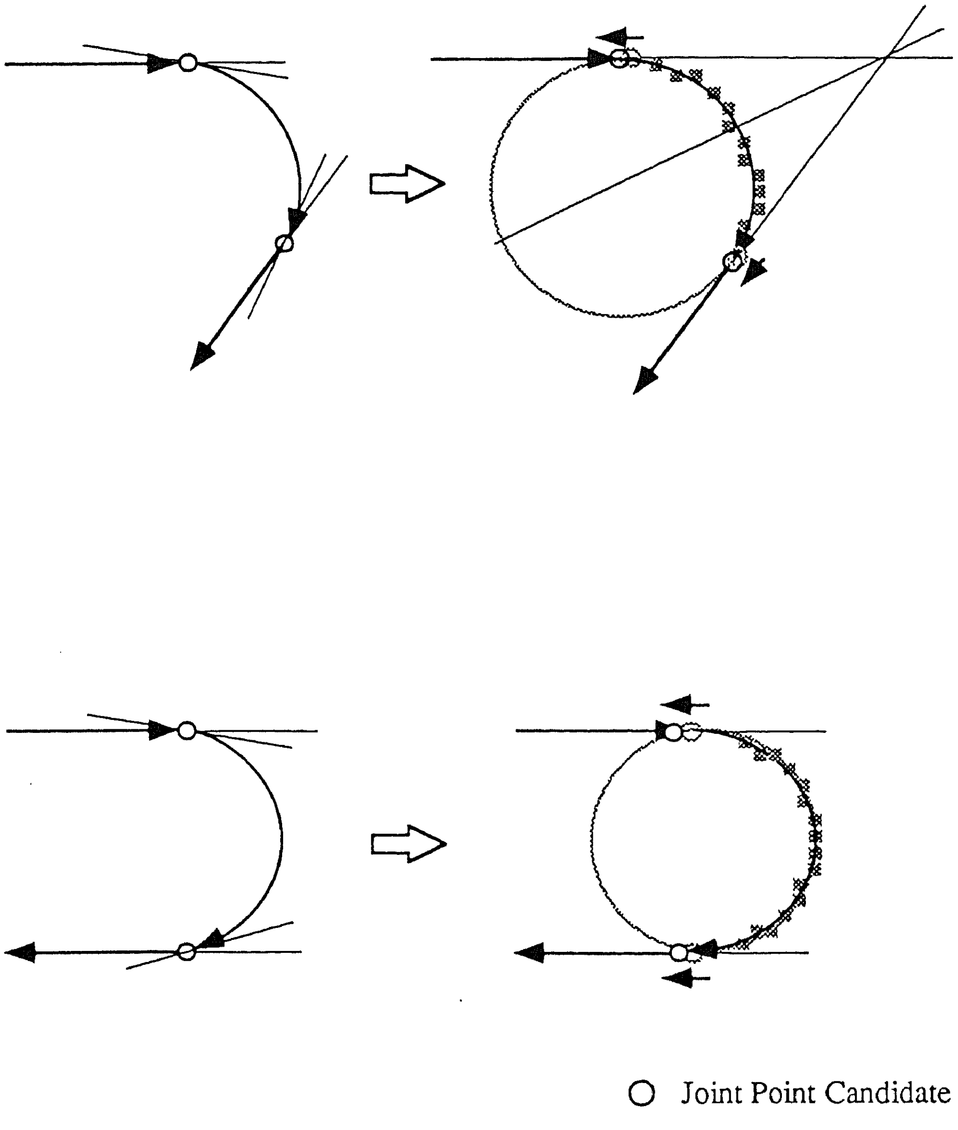


Figure 3.8: Connecting arc and line smoothly

In cases where this approximation fails, the set of boundary points is approximated with fluency function of degree 2. The approximation algorithm is just as same as that used in the calculation of the analog curvature. The dimension, coefficients $\{c_k\}_{k=-2}^{n-3}$, and the offset become stored data.

After all intervals are approximated, the parameters are stored as function-font data. Table 3.1 shows the stored data for line, arc and fluency function.

Table 3.1: Stored data

type	contents	size of data
straight line	flag for line	1 byte
	coordinate of next point	4 bytes
arc	flag for arc	1 byte
	direction of arc	1 bit
	coordinate of center point	4 bytes
	coordinate of end point	4 bytes
fluency function	dimension n	1 byte
	offset	4 bytes
	coefficients	$2n$ bytes
	# relative length of each knot interval	1 bytes

3.3.2 Reconstruction method

In this subsection, a method to reconstruct font typeface from function-font data is described.

A straight line is reconstructed by making a connection from the current point to the stored coordinate.

An arc is reconstructed by generating an arc which has a passes the current point and the stored end point with the stored center.

Other smooth line is reconstructed as follows. From the paper [102], the basis $N_{k,3}$ of the approximation function at sampling point $t_i \in [(T/n)(M-1), (T/n)M]$, is determined as

$$N_{k,3}(t_i) = \begin{cases} 0.5p^2 & , k = M \\ p(1-p)+0.5 & , k = M+1 \\ 1 - N_{M,3}(t_i) - N_{M+i,3}(t_i) & , k = M+2 \\ 0 & , k \leq M-1, M+3 \leq k \end{cases}$$

where $p = M - t_i \times n/T$. Then the approximation function is represented as follows.

$$s(t_i) = \sum_{k=M}^{M+2} c_k N_{k,3}(t_i)$$

3.4 Verification

In this section, the effectiveness of the system is verified.

Each module is implemented in the C language on a personal computer (NEC PC-9801DA, MS-DOS 3.3) as a separate application. After the image has been obtained as a bitmap image, each module for function-font are executed automatically. The effectiveness of the system has been verifyed evaluating the results of each module. Figures 3.9 and 3.10 shows how joint points candidates are extracted. Figure 3.11 shows the final result. The joint points are extracted properly at corners and at the junctions of straight line and arcs.

These modules are currently being transplanted by the authors in C++ on more powerful personal computer environment (NEC PC-9821 Ap, MS-DOS 5.0, MS-Windows 3.1). The purpose of this transplantation is to construct an integrated function-font generating system with interactive user interface. This makes it possible to get results with higher quality by revising the approximated results manually.

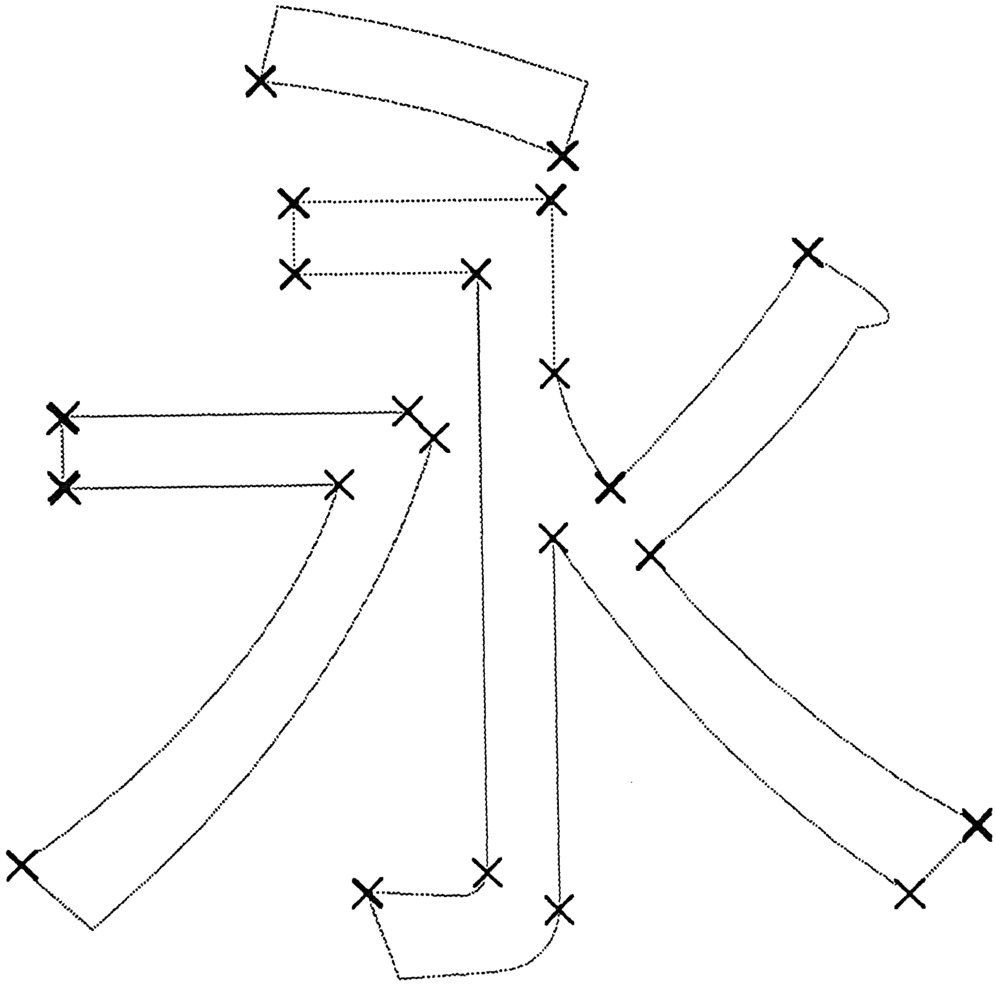


Figure 3.9: Joint points extracted based on digital curvature

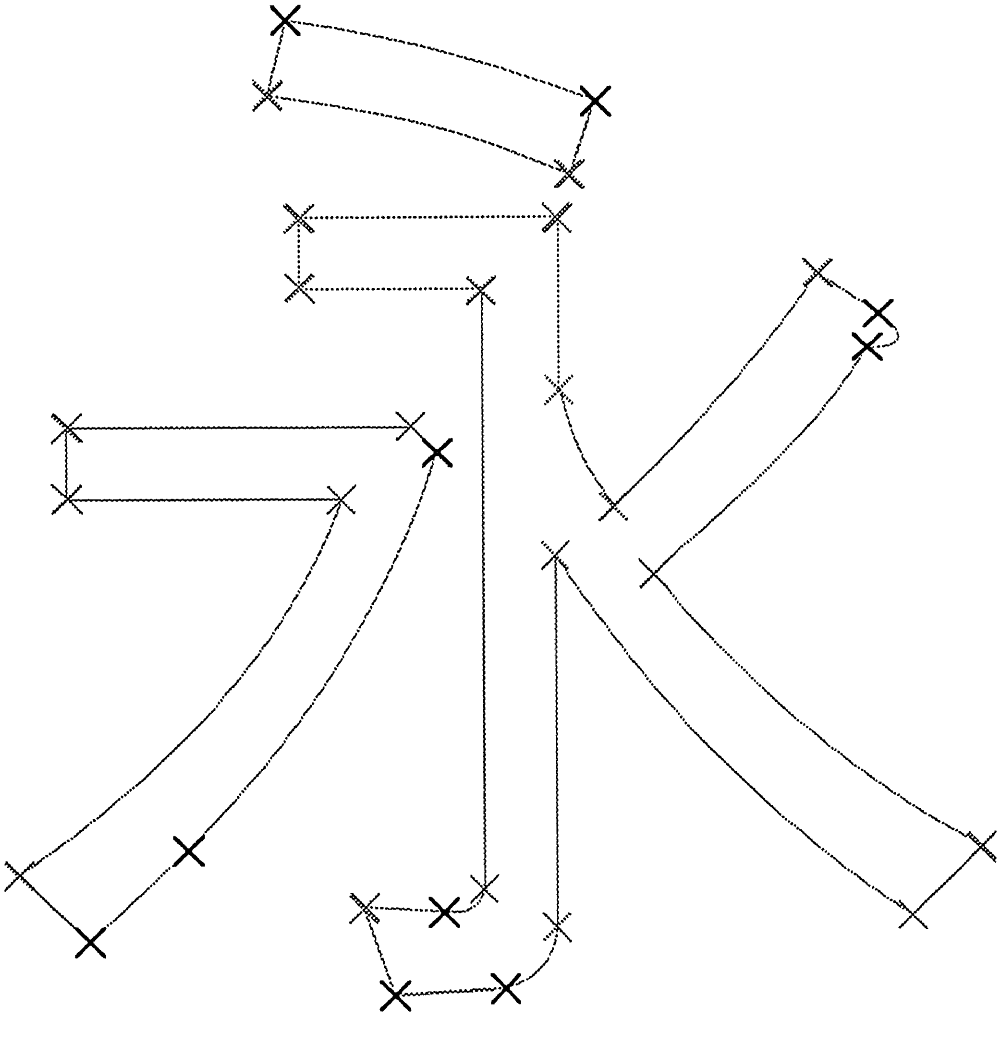


Figure 3.10: Joint points extracted based on analog curvature

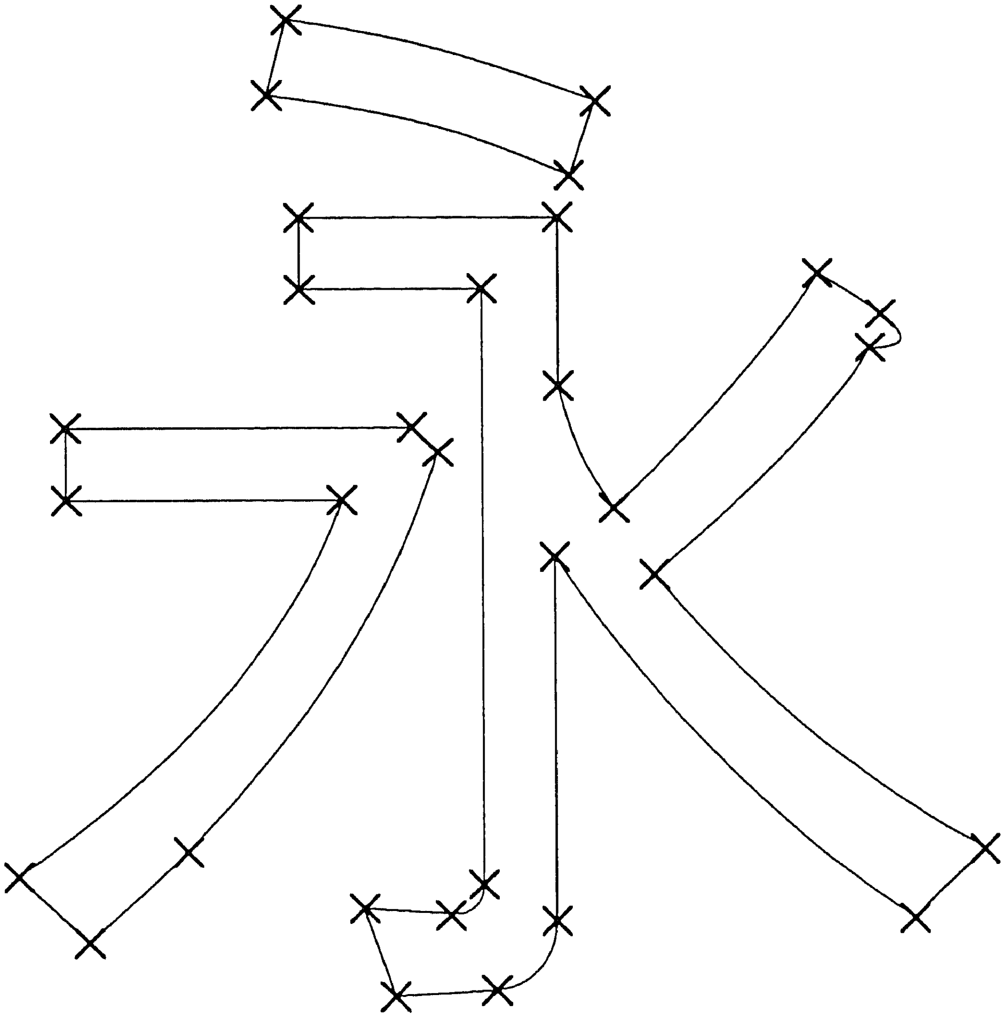


Figure 3.11: Reconstructed character

The system also provides printing modules which makes it possible to design an output image with various reshaping effects.

3.5 Summary

In this chapter, a practical system which automatically generates function-fonts for printed characters and figures has been proposed. The system provides various methods for noise reduction. To preserve the high quality of the original image, a multi-stage algorithm has been proposed to extract joint points properly. The effectiveness of the method was verified by constructing an actual function-font generating system.

Chapter 4

Compression Method for Grayscale Image

In this chapter, a compression method for grayscale image has been proposed as also an application for the compression method for a two-variable signal.

4.1 Introduction

As computational power has increased and made possible the development of large storage devices, an image archival system has come to be seen as a possibility. The system should store, retrieve, and display great many images according to the users' need. Since image data usually consumes large storage space in an uncompressed form, an image compression method is required.

As the previous research of the Wisdom Systems Lab, an image compression method for left ventricular cineangiograms[3] has been proposed. Left ventricular cineangiograms are high speed X-ray films of the left ventricle. In the proposed method, each cineangiogram frame is first divided into three regions using the statistical characteristics indicated in Ref.[119], and redundancy in the vertical direction of the image is reduced by a difference operation. Three regions are approximated by using a fluency function. The performance achieved through use of this method is evaluated on 20 examples of actual clinical data. The results show final results of about 7% of the original data quantity digitized at 1,000 × 1,000 [pixel/frame] and 10 [bit/pixel] while keeping an SNR of 40dB [p-p/rms].

The Wisdom Systems Laboratory is considering applying this method to other images. However, the compression method were specific for the left ventricular cineangiograms. We arranged the method so it can be used for other images. In this chapter, the performance of the arranged method is evaluated when when applied to grayscale images. A comparison is made with the JPEG compression scheme, which has been proposed as an international standard for still image compression.

The JPEG compression scheme is based on a two-dimensional discrete cosine transform (DCT) and Huffman entropy coding. The JPEG is a lossy compression method, i.e., the reconstructed image may differ from the original one. Lossy compression algorithms generally use aspects of the human visual system. For instance, the eye is much more receptive to fine detail in luminance signals than in chrominance signals. Also, the eye is less sensitive to energy with high spatial frequency than with low spatial frequency. Based on these aspects, the JPEG compression method codes the high-frequency coefficients with fewer bits. This produces distortion at edges with sharp contrast.

4.2 Specifications

In ths section, the specification in arranging the method is described.

In the previous method, from the requirement to reconstruct an image with quality as high as HDTV, the gray-level of the cineangiograms was quantized by 11 bits/pixel. However, a digital image handled with a computer usually only has 8 bits/pixel.

In the previous method, in order to achieve efficient compression, the image was divided into three regions based on the features of left ventricular cineangiograms.

The function approximation was performed to each raster line in the region, and the coefficients of the approximation function were stored directly as compressed data.

However, compression method which would be applied for various images cannot assume specific features for efficient coding. Therefore, in order to achieve a higher compression ratio, we consider encoding the coefficients of the approximate function by using Huffman entropy coding, which is also used in the JPEG compression scheme to encode the DCT coefficients. The image is divided into 8×8 pixel blocks in advance.

4.3 Compression and Reconstruction Method

In this section, a compression and reconstruction methods are designed in detail. Subsection 4.3.1 and 4.3.2 describes the compression and reconstruction method, respectively.

4.3.1 Compression method

In this subsection, the compression method is described in detail. The outline of the compression procedure is shown in Fig. 4.1.

First, the whole image is divided into blocks with size of 8×8 pixels. Each block contains 64 values which represents the gray-level of the pixel. The value varies from 0 to 255 according to the luminance. The width and height of the image are padded to be a multiple of 8.

Let $\{z(i,j)\}_{i=0}^{7}{}_{j=0}^{7}$ denote this original grayscale value. In the next step, the pixels in 8×8 blocks are re-ordered into a sequence. This is because we want to approximate the signal with one-variable functions. If two-variable functions are used, the calculation time increases because the number of parameters increases in comparison to the improvement in approximation precision.

The pixels are scanned in back-and-forth order so that each pixel comes from a

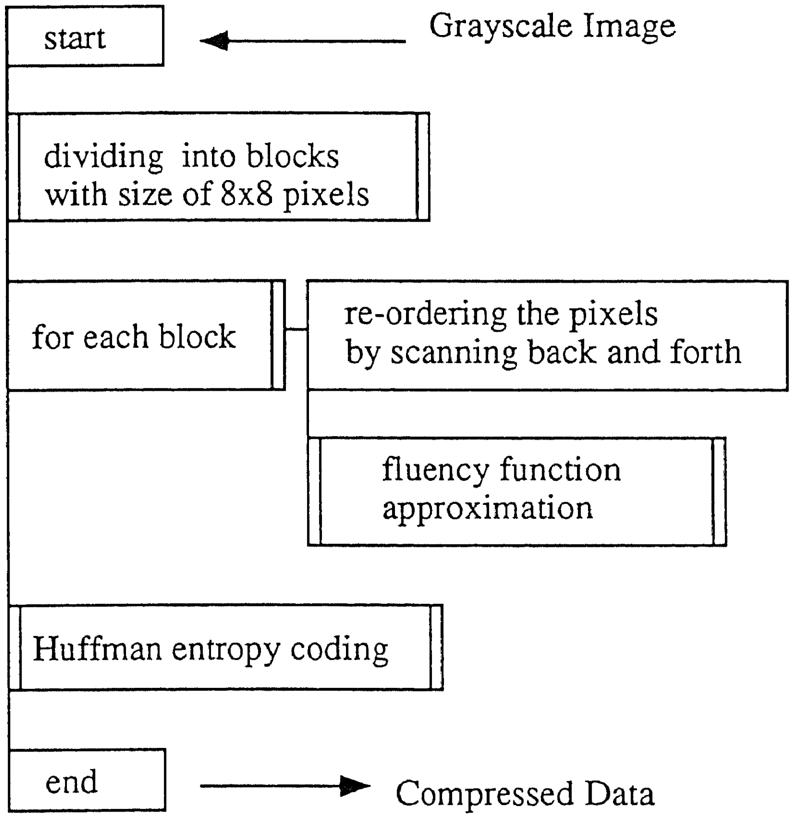


Figure 4.1: Compression procedure

neighboring location in the original block. This is based on the fact that neighboring pixels usually have high correlation. Let $\{w_k\}_{k=0}^{63}$ denote this sequence of signal. The signal $\{w_k\}_{k=0}^{63}$ is expected to have continuous gray-level variation. The signal can be approximated by a smooth function. Here we are using a fluency function[120] which gives less vibration in the approximated results.

In the case that the gray-level changes widely, the sequence is divided into some segments prior to the function approximation. The procedure for extracting feature points is as follows.

The first differences

$$\{d_k\} = w_k - w_{k-1}, \qquad k = 1, \ldots, 63$$

and the second differences

$$\{e_l\} = d_l - d_{l-1}, \qquad l = 2, \ldots, 63$$

are calculated. If $|e_l|$ is more than a threshold h, the point w_{l-1} becomes a feature point. The threshold h was determined as $h = 13$ acording to the experimental results. The functional approximation is performed for each segment. The SNR of the reconstructed method can be specified by giving the tolerance error of the approximation. The measure of approximation should be least mean square criterion. The most suitable dimension is determined by a repeating approximation while increasing the dimension.

In the following, the approximation algorithm for one segment is described. Let $\{v_i\}_{i=1}^{N}$ stand for the segment for convenience, and let $s(t)$ be an approximate function for the difference signal $\{v_i\}_{i=1}^{N}$. The approximation function $s(t)$ is represented by a piecewise polynomial function as follows[122].

$$s(t) = \sum_{k=-2}^{n-3} c_k \psi_k(t),$$

where $\psi_k(t)$ is a normalized time-limited function as

$$\psi_k(t) = 3(T/n)^{-2} \sum_{q=0}^{3} \frac{(-1)^q}{q!(3-q)!}(t - (T/n)(k+q))_+^2,$$

$$k = -2, -1, 0, 1, 2, ..., n-3,$$

and

$$(t-a)_+^2 = \begin{cases} (t-a)^2 & ,t > a \\ 0 & ,t \leq a \end{cases} .$$

The positive integer n refers the dimension of $s(t)$. The dimension n is decided by a repeating approximation as it increases by 1 until the approximation error falls below the specified tolerance error. The approximate function $s(t)$ is decided as follows. The normal equations

$$\sum_{k=-2}^{n-3} c_k \{ \sum_{i=1}^{N} \psi_k(t_i) \psi_m(t_i) \} = \sum_{i=1}^{N} v_i \psi_k(t_i),$$

$$m = -2, -1, ..., n-3$$

are to be solved for $\{c_k\}_{k=-2}^{n-3}$, to minimize the approximation error

$$Q = \sum_{i-1}^{N} (s(t_i) - v_i)^2.$$

The dimension n and the coefficients $\{c_k\}_{k=-2}^{n-3}$ are stored in one-dimensional array $\{w_l\}_{l=0}^{n}$. The remainder of the array $\{w_l\}_{l=n+1}^{63}$ is filled with zeros.

Finally, the sequences of array $\{w_l\}$ are encoded by Huffman entropy coding. The coded data are stored as compressed data.

The above process is illustrated in Fig. 4.2.

4.3.2 Reconstruction method

In this subsection, the reconstruction method is described. Figure 4.3 shows the reconstruction procedure.

First, the compressed data is decoded and the sequences of $\{w_l\}_{l=0}^{63}$ are obtained. For each block, the gray-level sequence $\{v_i\}_{i=0}^{63}$ is calculated from $\{w_l\}_{l=0}^{63}$. Since the component w_0 is the dimension and $\{w_l\}_{l=1}^{n}$ is the coefficients $\{c_k\}_{k=-2}^{n-3}$, the

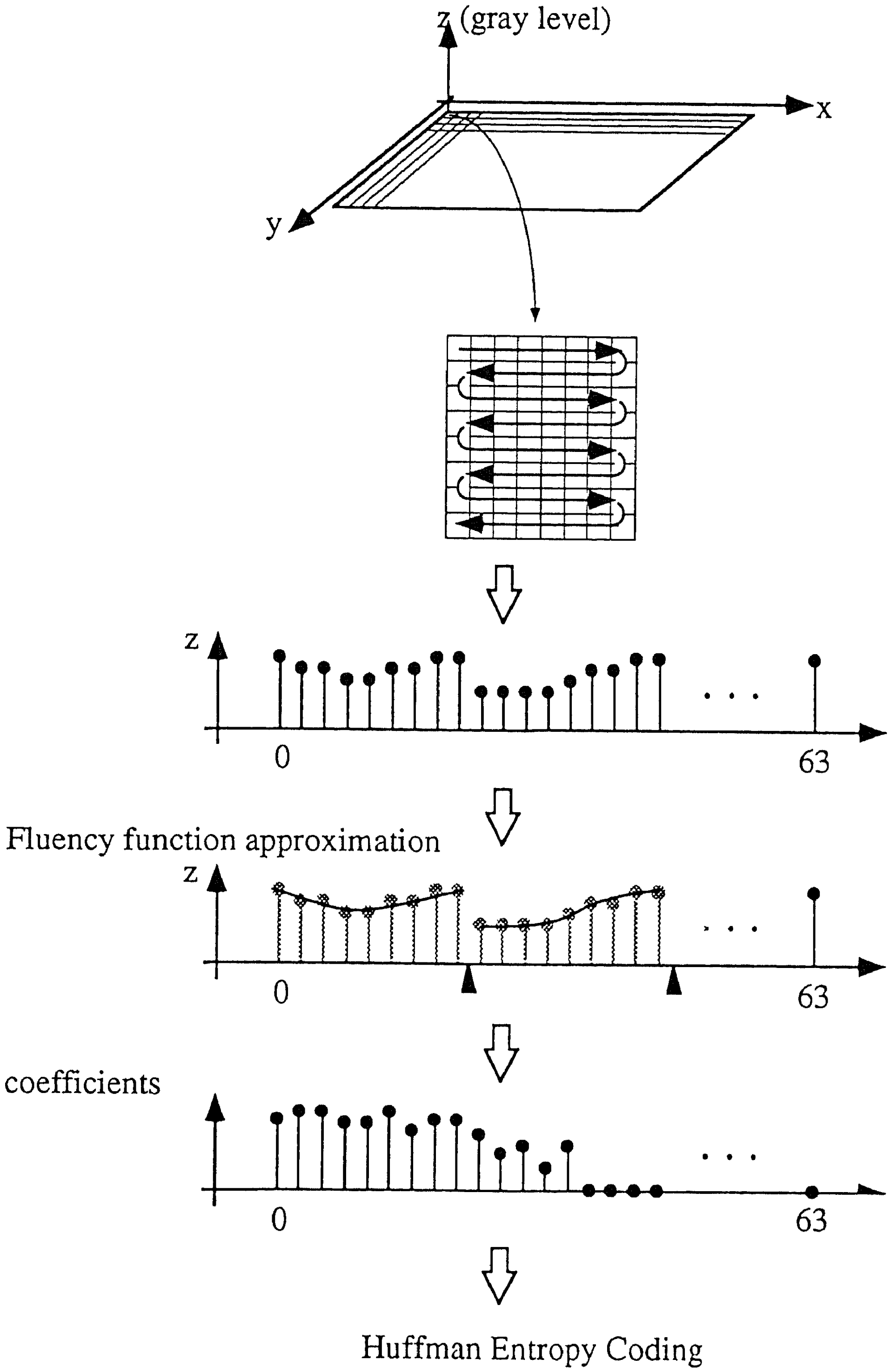


Figure 4.2: Coding process of the present method

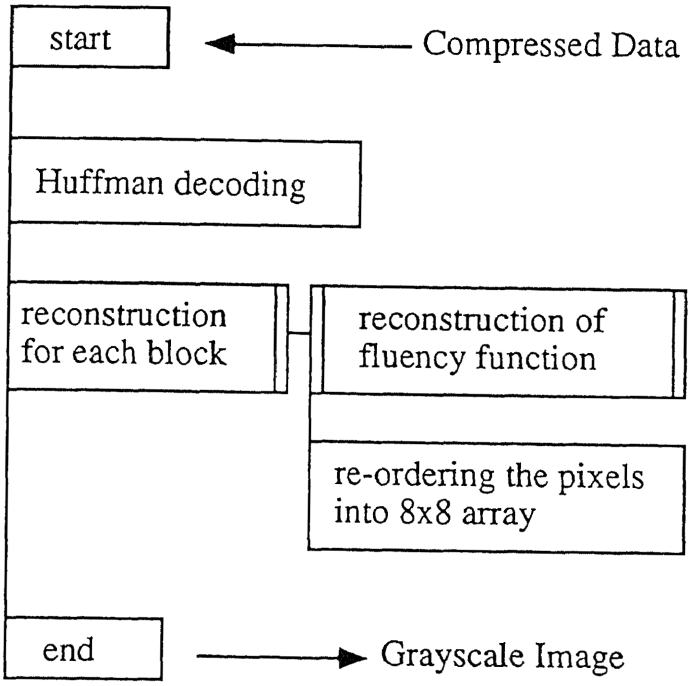

start
Compressed Data
Huffman decoding
reconstruction for each block
reconstruction of fluency function
re-ordering the pixels into 8x8 array
end
Grayscale Image

Figure 4.3: Reconstruction procedure

Table 4.1: Experimental Results

	proposed method	JPEG
data quantity	48Kbytes	65Kbytes
compression ratio	18.7%	25.4 %
compression time	153 sec	7 sec
reconstruction time	5 sec	7 sec
SNR	28.7dB	25.8dB

pixel map format. The data quantity of the image is 256Kbytes. Figure 4.4 shows the obtained grayscale image.

Figures 4.5 and 4.6 show the reconstructed image of the proposed method and that of the JPEG with default settings. Table 4.1 shows the size of compressed data, the compression ratio, compression and reconstruction time, and the SNR of the reconstructed image. The processing speed includes file i/o. Here, the SNR is defined as

$$(\mathrm{SNR}) = 10\log_{10}[256^2/\{\sum_{i=1}^{k_x}\sum_{j=1}^{k_y}(\hat{z}(x_i, y_j) - z(x_i, yj))^2\}][\mathrm{dB}]$$

where $\{z(x_i, y_j)\}_{i=1}^{k_x}$ and $\{\hat{z}(x_i, y_j)\}_{j=1}^{k_y}$ referes to the original image and the reconstructed one, respectively.

Table 4.1 shows the proposed method takes more compression time, but can obtain higher compression ratio and faster reconstruction speed.

4.5 Summary

In this chapter, a compression method for grayscale image has been proposed.

The method was designed by arranging the image compression method for left ventricular cineangiograms so that the method can be applied to an ordinary

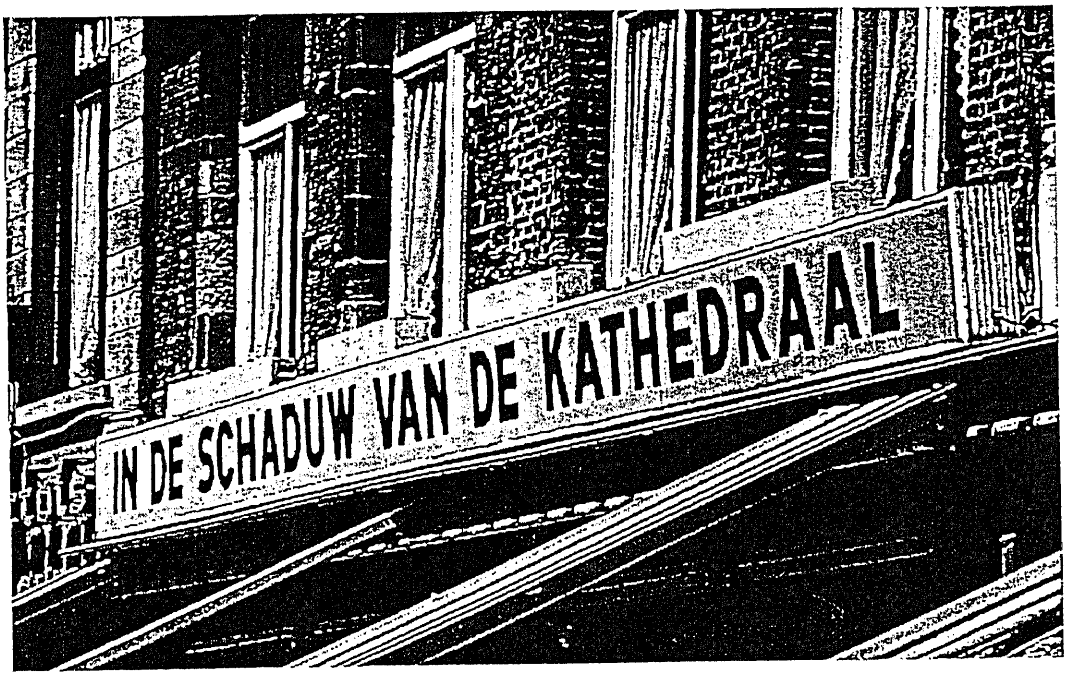


Figure 4.4: Original image

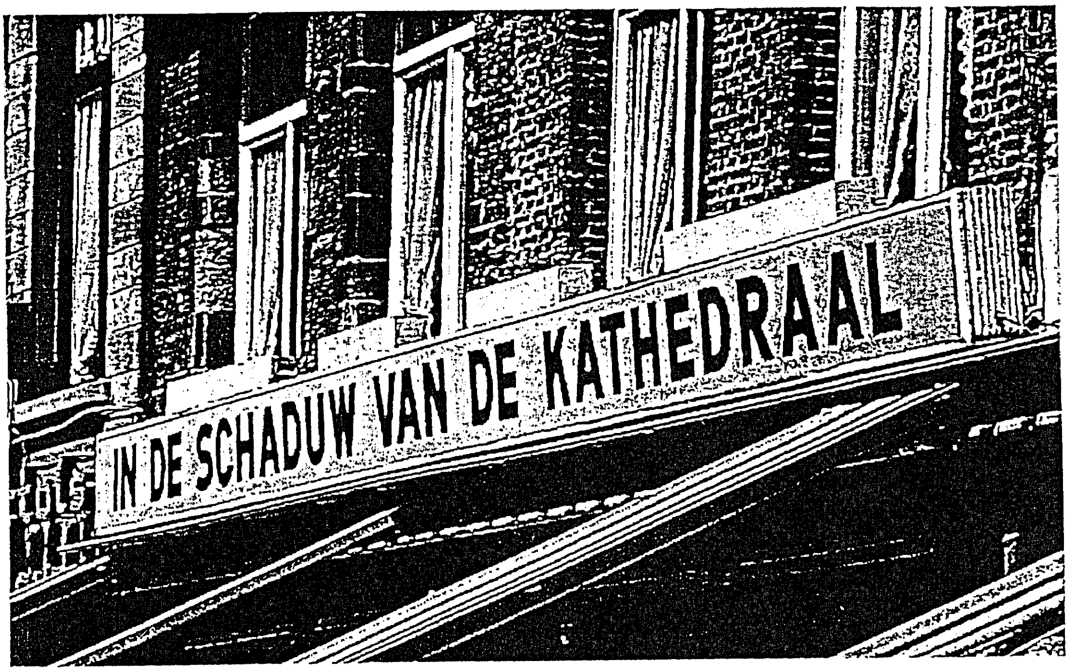


Figure 4.5: Reconstructed image by the proposed method.

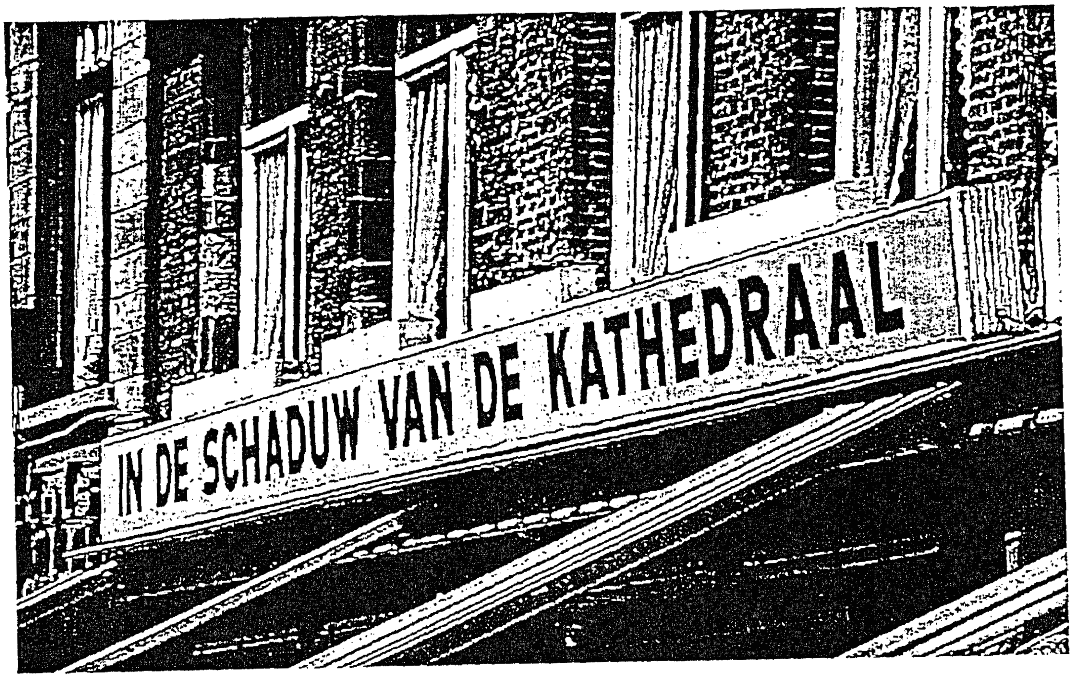


Figure 4.6: Reconstructed image by JPEG.

grayscale image. In this method, the image is first divided into blocks of 8×8 pixels in size. The pixels are then re-ordered into a one-dimensional signal by scanning the block in back-and-forth direction. The re-ordered gray signals are then approximated with a fluency function. The coefficients of the approximate function are then encoded by Huffman entropy coder. Since the cosine function can be represented as one of fluency function, the proposed method would be a generalization of DCT based compression method.

The performance of the method has been evaluated by applying it to an grayscale image from SCID. The method was implemented on a workstation by replacing the DCT operation in JPEG compression method with a fluency function approximation.

The results shows that although the method needed more compression time, it achieved a higher compression ratio. The reconstruction speed was slightly faster than the original JPEG compression scheme.

This method can be applied image archival systems, which need to compress the image data only once but have to transfer through communication line with a narrow bandwidth and reconstruct the image on a terminal which has only small processing power. A compression method for a full color image can be easily achieved by applying the method not only for the luminance signal but also for the color difference signals.

Conclusions

Chapter 5

Conclusions

In this chapter, the scientific and technical results which were obtained through Chapter 2 to Chapter 4 are summarized. Problems outside of the scope of the present study are also indicated.

5.1 Consideration of the Results of the Present Study

In Chapter 1, the rationale for the research is outlined.

When handling signals from nature with a computer, the signals are usually converted into discrete sampled values by an A-D conversion. However, this requires a large amount of storage. The signal can be characterized by the feature points and the behavior of the signal between them. The author tried to construct compression algorithms based on this principle. These algorithms extract the feature points which characterizes the signal and approximate the behavior of the signal by mathematical function. In this dissertation, fluency functions were used as approximation functions. The effectiveness of the algorithms have been proved by applying them to various kinds of signals.

In Chapter 2, a compression method for electroencephalogram was discussed as an application for a one-variable signal. In section 2.1, the background for the necessity of the compression system were described. In section 2.2, the specification for the compression method was determined from the requirement from clinicians. In section 2.3, the compression algorithm which satisfies the specification is proposed. Section 2.4 verifies the effectiveness of the proposed method. The experimental results show that the proposed method can compress the waveform into 64.8% of the original while keeping sufficient quality for diagnosis. Medical engineering usually requires high quality, and the performance of the function approximation satisfies clinicians.

In Chapter 3 and Chapter 4, compression methods for two-dimensional space were discussed as applications for two-variable signals.

In Chapter 3, a practical function-font generating system which generates a scalable font from a given bitmap image was discussed as an application of the compression method for a binary image.

In section 3.1, the background of this research and the problems which became clear through the initial researches are summarized. The problems which should be solved to construct a practical function-font generating system were also summarized. In section 3.2, the modules which constructs the system and their functions were described. In section 3.3, the algorithms to realize each module are described in detail. In section 3.4, the effectiveness of the system is verified by evaluating the performance of each module.

Through this investigation, it was found that the quality of the reconstructed image varies widely according to the location of feature points, and that noise reduction is required to extract these points properly.

Since it has become clear that computers are still a long way from demonstrating their full potential in the field of printing, both printers and computer researchers should cooperate and know both what is required and what is possible.

In Chapter 4, a compression method for grayscale digital images was discussed as an another application for two-variable signals.

In section 4.1, the background of this research was described. In section 4.2, the specifications which should be satisfied to construct an image archival system are determined. In section 4.3, a compression algorithm for grayscale image is proposed. The compression method extracts the points where gray-level changes without smoothness, and approximates the gray-level of pixels between them. The method is implemented by replacing the DCT operation in the international standard method for still image compression method with fluency function approximation. In section 4.4, the compression ratio, quality, and processing speed are compared with the international standard. It was found that the present method is superior in both quality of the reconstructed image and in reconstruction time.

5.2 Problems Left for Future Research

The verification results in each application show that the present algorithm is useful for representing various signals for a small quantity of data. However, there still remain many problems.

One of the most significant problems common to each compression method to do with the definition of the feature points. At the current stage, the feature points are those points where the behavior of the signal changes drastically. The feature points extracted in the proposed methods are based on knowledge provided by humans, which is implemented in the algorithm. A generalized rule should be developed to detect these feature points automatically.

Another problem is that it is not possible to know whether the feature points are extracted in the optimal place or not. This is because the meaning of optimal changes for applications. In the case of EEG compression, the optimal place is simply where stored data becomes minimal. However, in the case of automatic function-font generating system, the optimal place is where the function-font represents the characteristics of the ideal glyph of the typeface. The problem also concerns what kind of function is used to represent the behavior of the signal.

At the current stage of research, the optimal dimension of the approximation function is sought by iterating the approximation while changing the dimension and evaluating the error. This is a time-consuming algorithm and impairs real time processing in the compression stage. Therefore, an elegant way of finding the optimal dimension should be sought.

Acknowledgments

The author wishes to express his gratitude to his supervisor Professor Kazuo Toraichi Ph.D. of the Institute of Information Sciences and Electronics at University of Tsukuba for his wise direction and kindness. He is also grateful to Assistant Professor Naohisa Otsuka Ph.D. of the Institute for frequent stimulating and helpful discussions. The author would also like to express his gratitude to the following: Professor Naoto Sakamoto Ph.D. in the Institute, for his valuable comments and advice; Professor Kazuhiko Yamamoto Ph.D. in Electrotechnical Laboratory of MITI, for his valuable comments and encouragement; Professor Yoshihiko Ebihara Ph.D. in the Institute, for his helpful suggestions and discussions; Associate Professor Koichi Wada Ph.D. of the Institute, for his helpful suggestions and observations; Associate Professor Masaru Kamada Ph.D. in the Faculty of Engineering at Ibaraki University, for his continuing guidance and encouragement; Dr. Yoji Ishiyama in Toranomon Hospital, for his helpful advice and the permission to use his precious medical data for electroencephalogram; Dr. Minoru Ohyama Ph.D. at NTT, for his helpful discussion and the permission to also use his standard color image data. Finally, it is a pleasure to acknowledge the friendship and encouragement of the present members of the Wisdom Systems Laboratory, Institute of Information Sciences and Electronics, University of Tsukuba.

Author's Achievements

Papers in Journals

[1] K.Toraichi, Y.Ohtaki, Y.Ishiyama and R.Haruki: "Compressing EEG data using adaptive function approximation", *Transactions of the Institute of Electrical Engineers of Japan*, Vol.113-C, No.1, pp.14–20, 1993.

[2] T.Horiuchi, K.Toraichi, K.Yamamoto, H.Yamada and Y.Ohtaki: "On a method of restoring a polyhedron from three orthographic views by relaxation matching," *The Transactions of the Institute of Electronics, Information and Communication Engineers of Japan*, Vol.76D-II, No.1, pp.9–19 (Jan. 1993)

[3] K.Toraichi, T.Horiuchi, R.E.Kalman, Y.Ohtaki and H.Nagasaki: "Compressing Data Volume of Left Ventricular Cineangiograms," *IEEE Transactions on Biomedical Engineerings*, Vol.40, No.6, pp.579–588, 1993.

[4] T.Horiuchi, Y.Ohtaki and K.Toraichi: "On method of extracting joint points for generating function-fonts of multi-symbols," *Transactions of the Institute of Electrical Engineers of Japan*, Vol.113-C, No.12, pp.1136–1143 (Dec. 1993)

Papers at International Conferences

[5] Y.Ohtaki, K.Toraichi, T.Horiuchi and M.Kamada: "A method of automatically compressing fonts with high resolution," *Proceedings of the First International Conference on Document Analysis and Recognition*, Vol.1, pp.251–259, in St.Malo, FRANCE, 1991.

[6] Y.Ohtaki, K.Toraichi and Y.Ishiyama: "On Method of Compressing EEG Data for their Digital Databases," *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, Vol.4, pp.581–584, in SanFrancisco, U.S.A., 1992.

[7] Y.Ohtaki, K.Toraichi, T.Horiuchi and F.Nagasaki: "Data Compression Method of Left Ventricular Cineangiograms for their Digital Database System," *Proceedings of the IEEE Pacific Rim Conference on Computers, Communications and Signal Processings*, pp.602–605, in Victoria, CANADA, 1993.

[8] T.Horiuchi, K.Toraichi, Y.Ohtaki, K.Yamamoto and H.Yamada: "Restoring to a polyhedron from three orthographic views by relaxation matching method," *Proceedings of the IEEE Pacific Rim Conference on Computers, Communications and Signal Processings*, pp.626–629, in Victoria, CANADA, 1993.

[9] R.Haruki, K.Toraichi and Y.Ohtaki: "A multi-stage algorithm of extracting joint points for generating function-fonts," *Proceedings of the Second International Conference on Document Analysis and Recognition*, pp.31–34, in Tsukuba, JAPAN, 1993.

Papers under Submission for Publications

[10] K.Toraichi, Y.Ohtaki, Y.Ishiyama and R.Haruki: "A Compression Method for EEG Data Using Adaptive Function Approximation," *The EEG Journal*, (submitted for publication).

Reports at Symposiums and Technical Group Meetings

[11] R.Mori, M.Kawahara, M.Shinozaki and Y.Ohtaki: "Recent Status of "Superdistribution" Research Project," Proceedings of the Symposium on User Oriented Information Processing Systems, *Information Processing Society of Japan*, 1989 (in Japanese).

[12] S.Ueki, Y.Ohtaki and R.Mori: "The accounting process in Software Usage Monitor for Superdistribution," Paper on Technical Group Meetings, *Information Processing Society of Japan*, 90-IS-27-5, 1990 (in Japanese).

[13] R.Mori, M.Kawahara, M.Shinozaki, Y.Akiba and Y.Ohtaki: "Superdistribution Architecture," The Proceedings of the 1990 Symposium on Cryptography and Information Security, SCIS90-6a, *Institute of Electronics, Information and Communication Engineers of Japan*, 1991 (in Japanese).

[14] Y.Ohtaki and R.Mori: "Cryptographic managements for Superdistribution," Paper on Technical Group on Information Security, *Institute of Electronics, Information and Communication Engineers of Japan*, ISEC90-49, pp.33–42, 1991 (in Japanese).

References

[101] I.Yamasaki and H.Imura: "Curve-fitting for character outlines with circular arcs and straight lines," *Transactions of Information Processing Society of Japan*, Vol.26, No.4, pp.726–732, 1985 (in Japanese).

[102] K.Toraichi, K.Katagishi, I.Sekita and R.Mori: "Computational complexity for spline interpolation," *International Journal of Systems Science*, Vol.18, No.5, pp.945–954, 1987.

[103] T.N.E.Greville: "Spline Functions and Applications," Orientation Lecture Series No.8, MRC, University of Wisconsin, Madison, Wisconsin, 1970.

[104] E.L.Gibbs and T.J.Gibbs: "Microfiche data reduction for the EEG laboratory," *Clinical EEG*, Vol.14, No.2, p.106, 1983.

[105] R.D.Larsen, E.G.Crawford, P.W.Smith: "Reduced Spline Representations for EEG Signals," *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, p.804, 1977.

[106] W.Skrandies: "Data reduction of multi channel fields: global field power and principal component analysis," *Brain Topography*, 2/1–2, p.73, 1989.

[107] M.Kamada, K.Toraichi, R.Mori: "Periodic Spline Orthonormal Bases," *Journal of Approximation Theory*, Vol.55, No.1, p.27, 1988.

[108] K.Toraichi, I.Sekita, R.Mori: "On Automatic Compression of Fonts of High Quality Characters," *The Transactions of the Institute of Electronics, Information and Communication Engineers of Japan*, J70-D, p.1164, 1987, (In Japanese).

[109] I.Sekita, K.Toraichi, R.Mori: "On Effects of Various Conditions on Computational Complexity for Spline Interpolation," *The Transactions of the In-*

stitute of Electronics, Information and Communication Engineers of Japan, J70-D, p.1921, 1987, (In Japanese).

[110] Donald E. Knuth: "METAFONT : a system for alphabet design," Stanford Computer Science Report 762, Stanford, CA 1979.

[111] Donald E. Knuth: "TEXand METAFONT : New Directions in Typesetting," Digital Press, 1979.

[112] Donald E. Knuth: "The Metafont Book — Computers & Typesetting," Reading Mass, Addison Wesley, 1986.

[113] K.Shiraishi: "The history and design of characters," Kyushu University Press, 1984.

[114] "List of MORISAWA Type Face Fonts 1989," Morisawa, 1989.

[115] Microsoft Corporation: "True Type Font Files"

[116] H.Takahashi: "A Neural Net OCR Using Geometrical And Zonal-Pattern Features," *Proceedings of the First International Conference on Document Analysis and Recognition*, Vol.2, pp.821–828, 1991.

[117] "JPEG Technical Specification," JPEG8-R8, ISO/IEC JTC1/SC2/WG10, 1990.

[118] "Standard Color Image Data," Japanese Standards Association, 1989.

[119] K.Toraichi, K.Katagishi and R.Mori: "A left ventricular function analyzer and its application," *IEEE Transactions on Biomedical Engineerings*, Vol.BME-34, No.5, pp.317-328, 1987.

[120] K.Toraichi, I.Sekita and R.Mori: "An algorithm by hybrid splines," *International Journal of Systems Science*, Vol.19, No.8, pp.1547-1557, 1988.

[121] K.Toraichi, K.Katagishi, I.Sekita and R.Mori: "Computational complexity of spline interpolation," *International Journal of Systems Science*, Vol.18, No.5, pp.945-954, 1987.

[122] M.Kamada, K.Toraichi and R.Mori: "Periodic spline orthonormal bases," *Journal of Approximation Theory*, Vol.55, No.1, pp.27-38, 1988.