

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/38435306>

Optical music recognition using projections

Article

Source: OAI

CITATIONS

30

READS

224

1 author:



[Ichiro Fujinaga](#)

McGill University

152 PUBLICATIONS **1,960** CITATIONS

SEE PROFILE

Optical Music Recognition using Projections

by

Ichiro Fujinaga

Thesis submitted to
the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements
for the degree of
Master of Arts in Music Theory

Faculty of Music
McGill University, Montreal
September, 1988

© Ichiro Fujinaga, 1988

Table of Contents

Chapter 1	Introduction	1
Chapter 2	Background	4
	2.1 Non-optical input methods	4
	2.1.1 Alphanumeric	4
	2.1.2 Graphic	5
	2.1.3 Clavier and MIDI	5
	2.1.4 Combinations	6
	2.1.5 Digitized sound	6
	2.2 Previous research on OMR	6
	2.2.1 Prusslin and Prerau	7
	2.2.2 Others	8
	2.3 Applications	8
Chapter 3	Pattern recognition	10
	3.1 An Overview	10
	3.2 Projection	12
	3.3 Locating the maxima	15
	3.4 Syntax and semantics	16
Chapter 4	Music Printing and OMR	19
	4.1 Printing Methods	19
	4.1.1 Typography	19
	4.1.2 Engraving	20
	4.1.3 Lithography	23
	4.1.4 Modern Methods	23
	4.2 Music and OMR	26
	4.2.1 Orientation	27
	4.2.2 Shape and Size	28
	4.2.3 Positioning	29
	4.3 Conclusion	30
Chapter 5	Software Design	34
	5.1 Overview	34
	5.2 Locating the system	35
	5.3 Analysis of a system	43
	5.3.1 Locating the staff	43
	5.3.2 Locating the symbols	43
	5.3.3 Clef classification	45
	5.3.4 Key Signature Classification	46
	5.3.5 Classification of other symbols	46
	5.3.6 Beamed notes	50
	5.3.7 Determining the notehead position	51
	5.3.8 Locating the dot	51
Chapter 6	Experiment and conclusions	52
	6.1 Hardware and software	52
	6.2 Music samples and development	53
	6.3 Results	54
	6.4 Conclusions	62
Appendix		64
Bibliography		65

Abstract

This research examines the feasibility of implementing an optical music score recognition system on a microcomputer. Projection technique is the principal method employed in the recognition process, assisted by some of the structural rules governing musical notation. Musical examples, excerpted mostly from solo repertoire for monophonic instruments and representing various publishers, are used as samples to develop a computer program that recognizes a set of musical symbols. A final test of the system is undertaken, involving additional samples of monophonic music which were not used in the development stage. With these samples, an average recognition rate of 70% is attained without any operator intervention. On an IBM-AT-compatible microcomputer, the total processing time including the scanning operation is about two minutes per page.

Résumé

Cette recherche étudie la possibilité d'implanter un système de reconnaissance optique de partitions sur micro-ordinateur. La projection constitue la principale méthode d'analyse, quoique guidée par certaines règles de notation musicale. Un logiciel a été développé permettant de reconnaître les principaux symboles musicaux rencontrés dans des pièces publiées chez divers éditeurs et écrites, pour la plupart, pour instruments monodiques. Des exemples musicaux non utilisés à la phase du développement ont permis d'obtenir un taux de reconnaissance de 70%, et ce, sans intervention humaine. Le temps total d'analyse d'une partition, sur un micro-ordinateur de type IBM-AT, est d'environ deux minutes par page, incluant le temps de lecture optique (scanner).

Chapter 1

Introduction

Current vigorous developments in the areas of computer-assisted music composition and sound synthesis, together with successful recent designs of generative grammars for music production, have proved the applicability of computers to music. In one important area, however, there have been severe limitations: ever since its beginnings in the late 1940's, computer-assisted music score processing has been hampered by the lack of a fast and reliable system for computer recognition of music scores. Most research projects have resorted to time-consuming and error-prone hand methods of encoding musical notation. Experiments with optical scanners in the late sixties were successful in principle but never reached a stage where implementation would be practically or economically feasible. Recent progress in Japan and Korea involves expensive technology and is not expressly directed towards musicological research.

A practical and relatively inexpensive optical music recognition (OMR) system would allow a revitalization of computer-assisted research in musicology and, at the same time, simplify many tasks in music performance. Potential application areas include the establishment of large music databases for information

retrieval and research; score-based analysis of musical structure and style; score editing for reprint, revision, and preparation of performance materials; and re-coding for Braille printing.

The basic task of an OMR system is to convert the score into a machine-readable format by means of an optical scanner, the digitized image is then analyzed to locate and identify the musical symbols. Determining the feasibility of implementing an OMR program on a microcomputer with an inexpensive desktop optical scanner is the primary goal of the present research. One major difficulty here is that, in general, machine pattern recognition processes require large computer resources, for example, a single page of scanned music may contain more than one million bits of information. In order to analyze this data in a reasonable amount of time using a small computer, strategies must be devised to reduce computation time. The principal method proposed and investigated in the present thesis is the use of projections.

Projections essentially transform the two-dimensional scanned image into one-dimensional data, thus reducing the amount of data to be examined. The assumption is that, because of the distinctive features of each musical symbol, the reduced data retain sufficient information to locate and to help identify the target symbols. The use of projections for OMR has the additional advantage of minimizing the interference produced by the staff lines. Since most musical symbols are superimposed on the staff, separation of the individual symbol from its background usually poses some difficulties (Prusslin 1966; Prerau 1970). Projections have been used in other research, such as Chinese character recognition; nevertheless, extensive use of projections for music recognition has not been reported. To further reduce computation time, *a priori* structural knowledge of music notation is utilized.

Music scores rely on certain syntactical and semantic principles for effective communication. Some of these rules can be incorporated into the program to provide a more efficient and reliable recognition system. For example, the existence of a bar line is deducible from the total duration contributed by the notes and rests contained in the bar. Conversely, bar lines may be used as error-detecting devices. Because ex-

isting studies on the structural rules of notation as they apply to OMR are limited and fragmentary, only a small subset of these rules is utilized in the current program.

Since the present research is meant to serve as a pilot study, the current software implementation is subject to other limitations. The ability of the program to differentiate symbols is restricted to a subset of the symbols found in common musical notation. This subset includes clefs, accidentals, notes, and bar lines. While the section of the program that locates the staves on a page may be used for a wide variety of scores, the main portion of the program, devoted to identifying individual symbols, currently works only for music written for a single, monophonic instrument. The latter restriction is not critical since a complete OMR system will contain a number of subprograms, each specifically designed to analyze a certain type of score, furthermore, experience gained from monophonic scanning will be immediately applicable to the more complex situation found in polyphonic scores.

The details of the recognition techniques are explained in Chapter 3, various music printing methods as they affect OMR are studied in Chapter 4, and Chapter 5 contains a general description of the software. The results of software tests are given in Chapter 6. To provide some background, previous research on music input to computers and possible applications of the OMR system are reviewed in the next chapter.

Chapter 2

Background

2.1 Non-optical input methods

Although the potential of computers in the field of music has been recognized since the late forties, the difficulties associated with entering the music data into the machine have slowed the development of musical applications. Within the last thirty years, there have been several attempts to devise practical music input systems; the most important of these are reviewed below.

2.1.1 Alphanumeric

The earliest approach was to encode music notation into alphanumeric codes; currently, the most widely used among these appears to be Stefan Bauer-Mengelberg's "Digital Alternate Representation of Musical Scores" (DARMS; Erickson 1976). Development of the DARMS project has been slow since its inception; nevertheless, the system is capable of coding almost any type of standard musical notation. Two major disadvantages of alphanumeric encoding systems are that they are extremely time-consuming, and they are error-prone. A project of encoding 600 pages of Josquin Masses, undertaken at Princeton Univer-

sity and using an input language called Intermediary Musical Language, was estimated to have taken some 800 hours plus proofreading (Lockwood 1970, 20). Another drawback of these methods is that a certain amount of training is required to learn the encoding system. Other major alphanumeric encoding systems include Plaine and Easie code, MUSTRAN, Oxford Music Processor, and Leland Smith's MSS (Brook 1965; Brook 1970; Hewlett and Selfridge-Field 1987, 1–22; Smith 1973).

2.1.2 Graphic

The second method is graphic input, where the user selects the appropriate predefined music symbol from a menu shown on the computer screen, and places it on the staff using a pointing device such as a mouse. (Cantor 1971; Mercuri 1981a; Mercuri 1981b; Buxton et al. 1981; Yavelow 1985) This method works well if the music is relatively short and simple, otherwise the task can become very tedious. Yet, this is probably the most practical method for inputting any new music that has not been printed. Currently, there is a wide variety of commercial software available for microcomputers, including Professional Composer, Jim Miller's Personal Composer (Miller 1985), and Keith Hamel's Musprint and MusScribe. Hamel's programs contain a time-saving option, whereby the user can "draw" simplified musical symbols with the mouse. Many of these programs allow the user to play back the score through MIDI (Musical Instrument Digital Interface), which is an excellent means of error checking.

2.1.3 Clavier and MIDI

The use of a piano-like keyboard (clavier) attached to a computer offers an apparently efficient method of input (Raskin 1980a; Raskin 1980b; Talbot 1983). The user plays the music on the keyboard to transfer the information to the computer. With the advent of MIDI, input via instruments other than the keyboard, such as guitars and wind instruments, is now possible. One disadvantage of this method is the loss of some vital information such as chromatic orthography, slurs, stem directions, and voice assignments. The rhythm of the encoded music is often inaccurate since the process involves a human performer. In many systems of this type the user can specify the types of durational values contained in the music to minimize rhythmic errors.

2.1.4 Combinations

Some systems employ a combination of the methods already described. Smith's MSS, for instance, accepts alphanumeric codes which can be edited on the screen (Smith, 1973), and Xerox's Mockingbird system has clavier input but also allows interactive editing (Maxwell 1984). Much microcomputer MIDI-based software also has capabilities for modifying scores generated by MIDI input.

2.1.5 Digitized sound

The use of a digitized audio signal as input has also been attempted (Moore 1979; Piszczalski et al 1981; Chafe et al 1982; Foster et al 1982; Imai 1984; Piszczalski 1986). There are reports of reasonable success with monophonic music, but the task of decoding polyphonic music seems virtually impossible with present technology. Should this approach eventually give rise to a functioning method of score-conversion, the problems related to clavier input would still apply. Another disadvantage of this method is that it requires a powerful computer to process the immense amounts of data involved (about one million bits per second).

2.2 Previous research on OMR

A system involving an optical scanning device theoretically provides a complete, accurate and fast method for score input. The advantage of such a system was recognized as early as 1963 when Michael Kassler designed an hypothetical OMR machine named M (Kassler 1970). Unfortunately, due probably to the high price of scanners and a fairly small market, very little research followed this initiative. Recent dramatic reductions in the price of both computers and scanners, however have made the development of an OMR system an entirely realistic prospect, especially since this method offers many advantages over other types of input methods, and since there is a wide variety of applications which could benefit from these advantages. The users of an ideal OMR system would need no special training, complex human-readable encoding schemes need not be devised, and the system would be able to acquire all the information needed to reproduce the score faithfully.

2.2.1 Prusslin and Prerau

Despite the extraordinary potential of optical recognition of music, research into this area has been limited, until very recently, to two MIT doctoral dissertations: those by Prusslin (1966) and Prerau (1970). Because both were committed to the use of contour-tracing technique to segment the symbols, the staff lines became a considerable obstacle. Thus the bulk of their energy was devoted to solving this problem.

While Prusslin solved the staff line problem by removing all thin horizontal lines, his repertoire of symbols was limited to quarter-notes and beamed note groups. Also, the input to his system was restricted to samples consisting of one measure of relatively simple piano music.

Unfortunately, as Prerau found out, Prusslin's technique for removing the staff lines did not work when a larger set of symbols was to be recognized (Prerau 1970, 42). In order to isolate the symbols Prerau effectively removed the staff lines by contour-tracing the entire staff; this also meant erasing portions of the symbols that intersect with the staff lines. These "holes" were then filled in order to restore the musical symbols. After the segmentation process, the width and the height of symbols were used, along with musical syntax rules, for identification. Prerau chose Mozart's *Twelve Duets for Two Wind Instruments* K. 487, published by Breitkopf & Härtel, as his source music. Two to three measures of both parts were used as samples to develop the program which recognized clefs, certain time signatures, rests, accidentals, and notes. The music used for the test run consisted of a few samples (about twenty measures) taken from the same source and containing a total of 137 symbols, which the system correctly recognized. The time required to process each sample (equivalent to four to six measures of solo music) averaged about four minutes on an IBM mainframe. In a 1972 review of these two dissertations, Kassler remarked that:

... as a result from their work, the logic of a machine that "reads" multiple parallel staves bearing polylynear[sic] printed music in at least one "fount"[sic] and size can be seen to be no further than another couple of M.I.T. dissertation away. (Kassler 1972).

No such dissertation has appeared from M.I.T. or elsewhere.

2.2.2 Others

More recently, a few papers on computer recognition of music have appeared, originating from Japan (Ohteru 1985; Tojo 1982) and Korea (Lee 1985). The most impressive of these describes the Tsukuba Robot, which “reads” a page of keyboard music in about fifteen seconds, then performs it on an electronic organ using mechanical fingers and feet. The total development cost of the robot is estimated to be over two million dollars (Roads 1986).

The 1987 edition of the *Directory of Computer Assisted Research in Musicology* reports, without much detail, on research related to OMR conducted by Nicholas Carter (University of Surrey, Guildford, UK), Bernard Mont-Reynaud (Stanford University), Henry Baird (AT&T Bell Laboratories), Brad Rubenstein (Sun Microsystems and University of California, Berkeley), Peter Preston Thomas (University of Ottawa), Neil Martin (Thames Polytechnic, London), and Alastair Clarke (University of Cardiff) (Hewlett and Selfridge-Field 1987, 81-84).

2.3 Applications

Once the music scores have been stored in the computer, the data can be used by a wide range of applications. For musicologists and theorists, the computer can perform various useful and interesting tasks. These include score-based structure and style analysis, statistical validation of certain musical theories, creation of indices, thematic or otherwise; and the publication of reprints, revised editions, and critical editions. It is likely that the system will also encourage researchers to develop new analytical methods, applicable only with the assistance of computers.

There are also many possible applications for performers and conductors. Time-consuming tasks such as transposing music, creating parts from a score, making a piano reduction, or customizing scores for opera

production, would be facilitated by the use of computers. If the system reaches a point where it can recognize handwritten music, it can also assist composers in publishing high quality scores at a much lower cost than presently possible.

Perhaps the most important consequence of the present research is that the OMR system makes possible the establishment of large databases of music. Such databases would become an essential resource for most music research, including studies in perception and cognition. There is also an urgent need for an inexpensive method of transcribing ordinary music notation into Braille.

Chapter 3

Pattern recognition

3.1 An Overview

The general goal of image pattern recognition is to analyze a given image, which may consist of text, pictures, biomedical images, three-dimensional physical objects, or electrocardiograms, and recognize its content. There is no unifying theory available that can be applied to all kinds of pattern recognition problems, most techniques being problem oriented. The overall process can usually be divided into four stages: pre-processing, segmentation, feature extraction, and classification (figure 3.1).

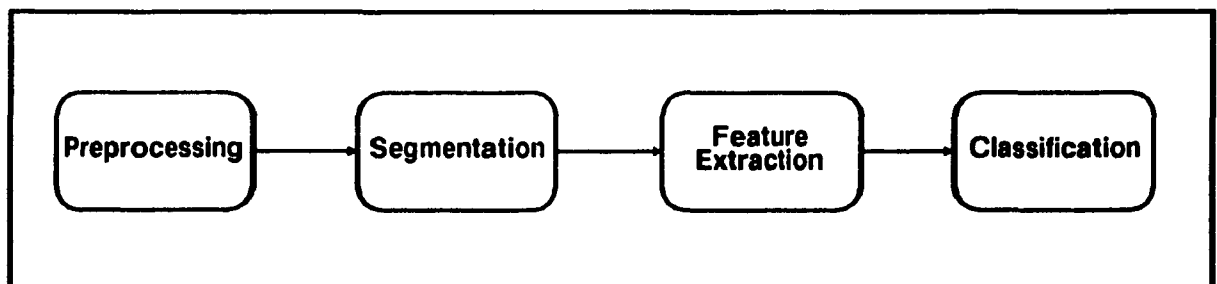


Figure 3.1 Overall image processing system.

Preprocessing involves the elimination of random noise, voids, bumps, isolated pixels, breaks, and other spurious components. Various equalization and filtering methods exist to perform the desired task. Although preprocessing is a standard procedure in other areas of image processing, it is excluded from the present recognition system in order to investigate the possibility of eliminating this step altogether on OMR systems. If this step can be eliminated without degrading the recognition capability, the processing speed of the system will be improved since preprocessing normally requires a large amount of computing time. The removal of the preprocessor is not totally unrealistic, for scores are read accurately by performing musicians. This suggests that the scores are sufficiently clear of blemishes, and there already is a high contrast between the musical symbols and the background.

Image segmentation locates and bounds certain areas that may contain the target objects. The separated components are then interpreted and recognized through higher-level processes. There are two major approaches to image segmentation: edge-based and region-based. In edge-based methods, local discontinuities (for example, a sudden change in the colour of the image) are detected first and then connected to form complete boundaries. In region-based methods, areas of the image that have homogeneous properties are found, these in turn give boundaries.

For each bounded area, a set of feature measurements and relations amongst these measurements are extracted to establish the distinctive properties of pattern classes. In certain applications, such as optical character recognition and OMR, it is best to extract those features which will enable the system to discriminate correctly one class of symbols from all other classes. Examples of features include physical measurements such as width and height, distribution of points, and symbol outline. In the present research projection technique is used both for segmentation (combining edge-based and region-based methods) and feature extraction.

Once the distinguishing features have been extracted, they are matched to a list of references or a knowledge-base for classification. In addition, other techniques may be used, such as distance measurements, shape deviation, shape matching, and hierarchical feature matching in the form of decision trees. Currently, a set of decision trees is implemented in the software using a predetermined set of features. Some theoretical background to the pattern recognition techniques used in the present research will be given next.

3.2 Projections

Projections are widely used for medical applications to reconstruct an object, such as the brain, using series of projections taken at different angles (Herman 1979). This reconstruction technique is also used in other fields, including radio astronomy (Bracewell 1979) and pollution control (Stuck 1977). In addition, projections have been used for shape analysis (Pavlidis 1977) and for segmentation, in particular for Chinese characters recognition (Nakao et al. 1973; Ogawa and Taniguchi 1979) and face recognition (Kanade 1977).

Since musical notation contains a relatively large, dark, and compact set of symbols of fixed size and orientation, projections become a powerful tool in the present recognition system. In addition, due to the insensitivity of projections to uniformly distributed random noise, the problem of isolating symbols from staff lines is minimized. Projections are used both for segmentation and for shape analysis of the symbols.

The generalized projection transform for the two-dimensional case is:

$$[Rg](s, \theta) = \int g(s \cos\theta - u \sin\theta, s \sin\theta + u \cos\theta) du.$$

The so-called Radon transform of $g(x, y)$ at (s, θ) is the integral of g along a line which passes through the point $(s \cos \theta, s \sin \theta)$ with slope $-\cot \theta$ (Herman 1979, 81–104). The two cases of special interest here are when θ is π and $\pi/2$, which give the projection onto a line parallel to the x - and the y -axis, respectively:

$$[Rg](s, 0) = \int g(s, u) du \quad \text{and} \quad [Rg](s, \pi/2) = \int g(u, s) du.$$

In the discrete case, given $P(i, j)$ of a $m \times n$ digital image, the equations above become

$$X(i) = \sum_{j=0}^n P(i, j), \quad 0 \leq i \leq m \quad \text{and} \quad Y(j) = \sum_{i=0}^m P(i, j) \quad 0 \leq j \leq n.$$

The two forms will be referred to as x -projection and y -projection, respectively. If the image contains only black and white pixels (bi-level), the projection is calculated simply by counting the number of black points along a certain direction. A program listing of a C-language function to calculate the x -projection of a rectangular bi-level image area is given in figure 3.2. See also figure 3.3 for a pictorial example.

```
void x_projection(
char**image,
int *xproj,
int  nw_row, int nw_col, /* top left corner */
int  se_row, int se_col) /* bottom right corner */
{
/* returns the projection onto the x axis of area defined by */
/* (nw_col, nw_row) and (se_col, se_row) inclusive of the image */
/* WARNING: no boundary check is performed */
int i = 0,
int row, col,

for (col = nw_col; col <= se_col; col++, i++)
{
xproj[i] = 0;
for (row = nw_row; row <= se_row; row++)
if (!image[row][col]) /* 0 = black, 1 = white */
xproj[i]++;
}
}
```

Figure 3.2 X-projection function

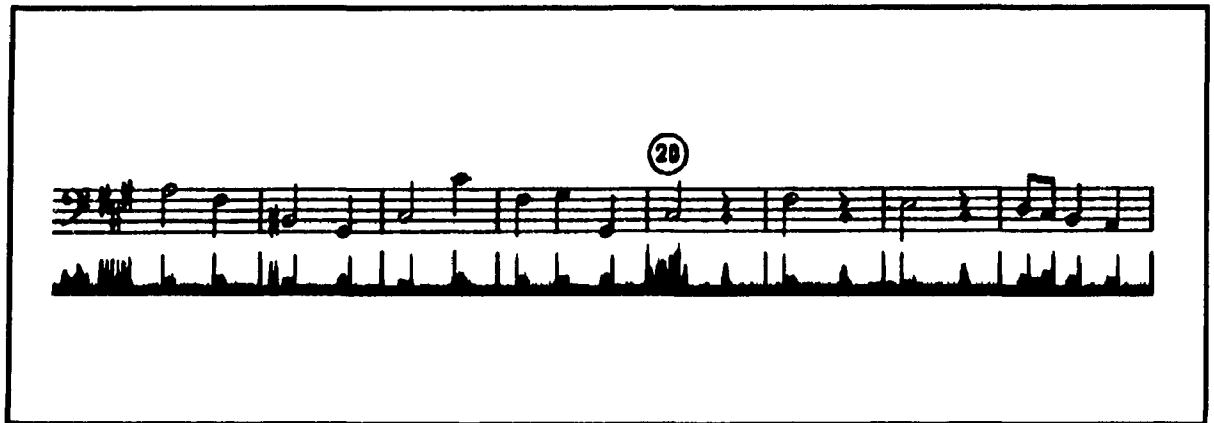


Figure 3.3. An example of x-projection

While only these two values of θ are used in this work, other values can be employed. Nakao (1973), for example, uses diagonal projections, with $\theta = \pi/4$ and $3\pi/4$. Another variation of the technique, though not implemented, is to keep track of the discontinuities or the holes in the picture when taking the projection, as investigated by Pavlidis (1977, 150–54).

After the projection is obtained, it must be analyzed to extract useful information. There are many methods available for this process since it falls within the domain of waveshape analysis (Pavlidis 1980). Certain features such as width, height, and area of the projection profiles (which can be easily calculated), are used to determine the presence and position of a symbol, and also to narrow the choices in identifying the target symbol. To obtain other features, a Fourier transform can be taken to examine the amplitude spectra (Nakano et al. 1973) or derivatives can be calculated (Levine and Leemet 1975). The former technique is not implemented because of the computation time required. The derivative, however, is used extensively to find the maxima in the projection.

3.3 Locating the maxima

Two steps are required to locate the maxima. First, the derivative is calculated, then the points where the resulting function intersects the x-axis, called the zero-crossings, are determined. To obtain the derivative a simple approximate differentiation formula,

$$Df(x) = [f(x+1) - f(x-1)] / 2,$$

which is widely used in practice, was found to be sufficient for the present needs. Next, the slope function is scanned for a change in the sign. Since this method will locate both maxima and minima, an algorithm was devised so that only those zero-crossings whose original functional value exceeded some threshold value were retained (See figure 3.4). The suitable threshold values were determined by trial and error.

```
int zerox_max(
int *slope, /* peaks are stored here */
int *proj, /* the original projection */
int size, /* the size of the array */
int min) /* minimum value for the maxima */
{
/* find zero crossings of maxima only */
/* returns # of peaks, indices stored in slope[] */

int i, j;
int count = 0, /* # of maxima */
int *temp, /* stores indices to proj[] of maxima */
temp = malloc_int(size / 2); /* assume no more than size / 2 maxima */

size--;
for (i = 0; i < size; i++) /* find zero crossings */
{
for (j = i + 1; slope[j] == 0 && j < size; j++) /* skip consecutive 0's */
;
if (slope[i] > 0 && slope[j] <= 0 && (proj[i] > min || proj[i + 1] > min))
temp[count++] = i;
}
for (i = 0; i < count; i++) /* store the results in slope[] */
slope[i] = temp[i],
free(temp),
return(count); /* # of peaks */
}
```

Figure 3.4. Zero-crossing function

3.4 Syntax and semantics

The task of pattern classification may be expressed as finding a mapping from input patterns onto possible pattern classes. One way to simplify the process is to reduce the number of possible pattern classes. Syntactical information is used for this purpose.

By using some basic definitions of formal grammars and languages, a clear representation of a music notation system can be established.

A grammar is a 4-tuple $G = (N, T, P, S)$,

where N is a finite set of non-terminals

T is a finite set of terminals ($N \cap T = \emptyset$)

P is the set of the finite number of productions of the form $a \rightarrow b$;
where $a \in V^*NV^*$, $b \in V^*$, $V = N \cup T$ and $V^* = V \cup \{\lambda\}$, λ is the empty string
(the symbol \rightarrow means “can be replaced by”)

$S \in N$ is the start symbol (Fu 1982, 53–54).

The language generated by a grammar G , denoted $L(G)$, is the set of sentences generated by G .

$L(G) = \{w \mid w \in T \text{ and } w \text{ can be derived from } S \text{ by applying one or more productions from } G\}$.

A context-free grammar has productions of the form:

$A \rightarrow b$ where $A \in N$ and $b \in V^* - \{\lambda\}$.

Note that the replacement of the non-terminal A by the string b is independent of the context in which the A appears (Fu 1982, 55). Furthermore, a context-free grammar is said to be $LL(k)$ if the top-down parser can be made to work deterministically by looking at k input symbols beyond its current position (Aho and Ullman 1972; Fu 1982, 180–82). Music notation grammar is context-free and $LL(k)$; this is in effect what allows musicians (top-down parsers) to read the music as efficiently as they do.

As an illustration of how formal grammars can be applied to OMR, let $\{b, d, f, g, h, w\}$ represent the possible pattern classes for a sequence of input patterns $\{X(1), X(2), \dots, X(n)\}$. Note that for each $X(i)$, there are six possible choices. Now define

$$\begin{aligned} G &= (N, T, P, S), \text{ where} \\ N &= \{S, N, W, C\}, \\ T &= \{b, d, f, g, h, w\}, \\ \text{and } P: \quad S &\rightarrow CNb \quad N \rightarrow NN \mid W \mid h \\ \quad \quad W &\rightarrow w \mid wd \quad C \rightarrow g \mid f. \end{aligned}$$

(Replace C with clef, N with note, W and w with whole note, d with dot, h with half note, g with soprano clef, f with bass clef, and b with bar line, then the example can be easily understood to represent a very simplified grammar for an opening bar of music.)

Using G, assume $X(1) = g \mid f$, and $X(2) = w \mid h$. If $X(3)$ is found to be

- 1) w, then $X(4) = w \mid d \mid h \mid b$; or
- 2) h or d, then $X(4) = w \mid h \mid b$.

Thus G helps to limit the number of pattern classes to a maximum of four instead of the original six.

The reduction process can be augmented by another property of languages, namely, semantics. The motivation for developing a semantics-based technique arises from some of the limitations of a purely syntactic approach where context is not taken into account (Baird and Kelly 1974), and the fact that not all structural information can be easily put into the symbol-oriented form of the productions. The injection of semantic considerations into a context-free grammar is called semantic grammar (Tang and Huang 1979), or attributed grammar (Fu 1982, 116–23). The idea is to reduce the large number of possible choices derived from production rules by employing some higher-level rules which govern a particular language. Consider, for example, a sentence that begins with “The cat climbed up the ...” A simple grammar may allow any noun to follow the second article. One way to limit the number of possible nouns is to apply some contextual rules.

One piece of semantic information useful for OMR is the rule which states that the durational values of all notes and rests within a bar must add up to the value indicated by the meter signature. For example, a 4/4

bar must contain four quarter notes or their equivalent. (There are exceptions to this rule, especially in pre-Baroque music.) Using the grammar G above, define $var(x)$ to indicate the durational value of symbol x and let $var(w) = 1.0$, $var(h) = 0.5$, $var(wd) = 1.5$, and $var(C) = 0.0$. Assume that an input string is partially identified as $\{gwwd, X(6), X(7), \dots\}$ and let $var(S) = 4.0$, where S denotes the sequence of symbols allowed between bar lines. Since $var(gwwd) = 3.5$, it follows, by process of elimination, that $X(6) = h$ and $X(7) = b$. Thus the rule provides the system with information about the existence of both 'h' and 'b' in advance, requiring no further analysis.

In order to implement successfully the recognition techniques discussed in this chapter, it is necessary to analyze carefully the objects to be recognized. The effects of music printing methods on pattern recognition are investigated in the next chapter.

Chapter 4

Music Printing and OMR

4.1 Printing Methods

Music printing began in the late fifteenth century. Typography, a method that uses hundreds of small moveable types, was the most common printing technique throughout most of the sixteenth and seventeenth centuries. Engraved copper plates became the preferred printing technique during the eighteenth century. In the early 1800s, lithography was introduced and adapted to music printing. These were, until the twentieth century, the most common methods of printing music.

4.1.1 Typography

Typography involves combining a small block of metal or wood, called type, with a raised letter or symbol. When inked and pressed on paper, the symbol leaves a printed impression (see figure 4.1). Using type to set music is problematic because many small pieces of type must be combined to give the impression that the musical symbols are overlaid on top of continuous staff lines. This often results in broken staff lines, as shown in figure 4.1. Despite the inferior quality of print, typesetting was used well into the twentieth century

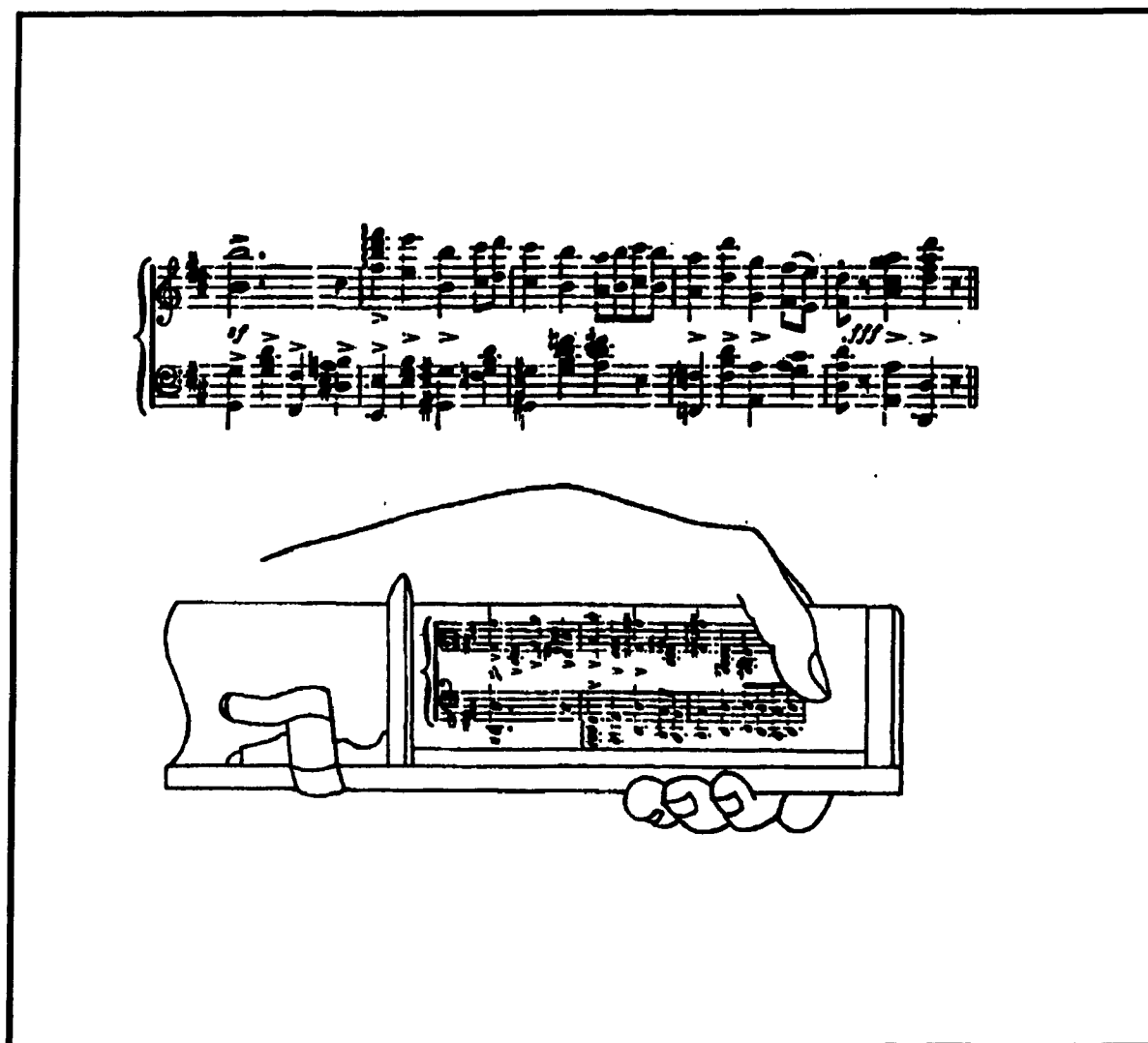


Figure 4.1 Example of music typesetting (Ross 1970, 11)

because it facilitated the combination of music and text for psalm books, hymnbooks, and music text books (Poole 1980, 247). (See figure 4.2 for a set of nineteenth-century types.)

4.1.2 Engraving

Engraving is a technique of producing prints by making cuts or indentations in a metal plate. When this technique was first developed, around 1550, all symbols were cut freehandedly (see figure 4.3). In the early eighteenth century, punches became available to engrave the more important music symbols, such as note-

Synopsis of Characters in the Gem Music.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	00
101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192	193	194	195	196	197	198	199	200
201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	256	257	258	259	260	261	262	263	264	265	266	267	268	269	270	271	272	273	274	275	276	277	278	279	280	281	282	283	284	285	286	287	288	289	290	291	292	293	294	295	296	297	298	299	300
301	302	303	304	305	306	307	308	309	310	311	312	313	314	315	316	317	318	319	320	321	322	323	324	325	326	327	328	329	330	331	332	333	334	335	336	337	338	339	340	341	342	343	344	345	346	347	348	349	350	351	352	353	354	355	356	357	358	359	360	361	362	363	364	365	366	367	368	369	370	371	372	373	374	375	376	377	378	379	380	381	382	383	384	385	386	387	388	389	390	391	392	393	394	395	396	397	398	399	400
401	402	403	404	405	406	407	408	409	410	411	412	413	414	415	416	417	418	419	420	421	422	423	424	425	426	427	428	429	430	431	432	433	434	435	436	437	438	439	440	441	442	443	444	445	446	447	448	449	450	451	452	453	454	455	456	457	458	459	460	461	462	463	464	465	466	467	468	469	470	471	472	473	474	475	476	477	478	479	480	481	482	483	484	485	486	487	488	489	490	491	492	493	494	495	496	497	498	499	500

Figure 4.2. Set of types (Gamble [1923] 1971, 179)

The image displays three systems of early engraved musical notation, likely from a 17th-century manuscript. Each system consists of three staves. The notation is a form of early musical shorthand, with notes represented by various symbols (dots, lines, and curves) placed on and around the staves. The lyrics are written in Italian and are placed between the staves.

System 1:

Ornamento uol = go. rium = que gi mi passi Par, che di me si pianga
Ornamento uol = go. rium gi = rei passi Par, che di me si pianga ari, sa =

System 2:

si sos. pi. ri, Par, che di me si pianga Par, che di me si pianga, e si sos. pi. ri
si = ri Par, che di me si pianga si pianga, e si sos. pi. ri

System 3:

Par, che di ca ciascuno messo al mio duolo, Che fui tu qui me =
Par, che di ca ciascuno messo al mio duolo, Che fui tu qui me = schin

Figure 4.3. Early engraved music (1613?) (Krummel 1975, 147).

heads and accidentals (see figure 4.4). A set of punches for a given staff size usually consists of about fifty punches. Figure 4.5 shows sets of punches of different sizes. Symbols not represented in the set are still cut by hand, including stems, beams, barlines, ledger lines, and ties. Text is struck with a set of alphabet punches. “Punches for such indications as *cresc.*, *dim.*, *rall.*, etc., and for *ff*, *pp*, *mf* are obtainable with all the letters on one punch, but these are not much used now, each letter being struck separately.” (Gamble [1923] 1971, 140)

4.1.3 Lithography

Lithography is a printing process where the design, drawn on a flat surface, is treated to retain ink while the non-image areas are treated to repel ink. At first, the music was drawn freehandedly using special ink, but by the 1830s there were some special devices which deposited ink in the shape of the musical symbols; noteheads, for example, could be printed this way.

4.1.4 Modern Methods

With the introduction of the camera, it became possible to print music from an original written on ordinary paper. To improve the quality of print, several methods were devised to draw symbols on the page. One of these involves stamping the paper with an inked steel punch, applying pressure by hand (see figure 4.6). A method using stencils, called the Halstan process, is used for the musical examples of the *New Grove Dictionary of Music and Musicians*, and by Faber Music (Poole 1980, 258). Rub-off transfer sheets is yet another method used for printing music; according to Poole, it is “extensively used by Bärenreiter.” (Poole 1980, 258) Several types of music typewriters were also invented.

Since the 1970s, computers have been used increasingly to print music. In the 1980s, several commercial computer programs became available to print music with computer printers or plotters; some have the ability to send their output to phototypesetters. Some of the major music publishers using this new technology are Belwin Mills, Bärenreiter, and Oxford University Press (Hewlett and Selfridge-Field 1987, 293–4).

16

VIOLINO PRIMO CONCERTINO

CONCERTO I V

Larghetto. Affettuoso

Adagio

Allegro

This image shows a page of handwritten musical notation for a violin concerto. The page is numbered '16' in the top left. The title 'VIOLINO PRIMO CONCERTINO' is centered at the top, followed by 'CONCERTO I V'. The notation is written on ten staves. The first staff begins with a treble clef and a key signature of one sharp (F#). The tempo marking '*Larghetto. Affettuoso*' is written above the second staff. The tempo marking '*Adagio*' appears above the eighth staff. The tempo marking '*Allegro*' is written below the ninth staff. The notation includes various musical symbols such as notes, rests, and slurs, all rendered in a handwritten style.

Figure 4.4 Music engraved with punches (Handel 1746, 16)

No. 1

No. 3

No. 4

No. 5

No. 6

No. 7

No. 8

Figure 4.5. Set of punches of different size (Ross 1970, 21–2)



Figure 4.6. Examples of stamped music (Ross 1970, 35)

4.2 Music and OMR

Despite the many methods available to print music, for the purpose of OMR all symbols fall in one of two categories: those formed entirely by hand and those produced by some kind of tool or font. These tools may be engraving punches, moveable types, dies on a music typewriter, symbols on a rub-off sheet, stencils, or computer-generated fonts. Orientation, shape, size, and positioning are a symbol's key features; the de-

degree of uniformity in these features depends on the choice of printing method. The next section examines how and to what extent the symbols' features are affected by printing, and how this in turn affects OMR.

4.2.1 Orientation

Most musical symbols are superimposed on the staff, therefore the OMR process must first determine the orientation of the staves themselves. Often, the staves are not parallel to the paper's top edge; in some cases, the staves on a page may not be parallel to each other. This is usually due to each staff being ruled manually, using a T-square. Carelessness during this operation results in non-parallel staves. The engravers possess a tool called the scorer, which has five evenly spaced teeth, to rule the staves. However "many engravers rule each individual line;" (Ross 1970, 70) "they say that they dislike the five-point tool because it requires more force for ruling the five lines at one time and then it is often necessary to re-engrave some lines that are badly ruled." (Gamble [1923] 1971, 93). If such is the case, there is even a possibility that the lines within a staff may not be parallel to each other.

Since ruling the staves and placing the symbols on the staff are two unrelated steps, the orientation of the symbols with respect to the staff is seldom perfect. Normally, the lines are ruled first, then the symbols are placed individually on the staff. No guide other than a pair of trained eyes can ensure their proper orientation. There are a few exceptions. Some music typewriters have the capability to rule the staff lines, thus minimizing errors, yet there is always the possibility of slippage in the paper feeding mechanism. Note that, owing to the nature of musical notation, there are far more vertical movements of the paper while typing music than while typing ordinary text. On movable types, the appropriate portion of the staff is attached to the symbol, ensuring a fixed relationship between the staff and each symbol. In this case, the symbols will be oriented correctly provided that the types were cast properly. The only process that guarantees correct orientation is computer printing, where the machine "knows" the exact position of both the staff and symbols.



Figure 4.7. Various forms of treble clef

4.2.2 Shape and Size

Although the shape and size of musical symbols have remained relatively constant since the eighteenth century, there are some differences depending on where and when the score was printed. (Figure 4.7 shows a sample of different treble clefs that can be found in modern editions by various publishers.) Moreover, a single publisher may choose to use different fonts or methods of printing as the company evolves (see figure 4.8).

Variations in size and shape can be found within a single page, even when printed using fixed “tools.” Symbols placed by types should be uniform, since all types are cast from a single mold. These fragile, small, finecut types are often broken or chipped, however. When engraving punches are used, irregularities may be caused by varying depths of indentation, which “must not be more than 1/64 inch.” (Gamble [1932], 1973) For example, the two whole notes on the top two staves in figure 4.5 were presumably engraved with the same punch, yet the top one is 3.5 mm wide and the other 4 mm wide.

Obviously, symbols cut or drawn by hand vary even more in size and shape, the degree of consistency being dependent upon the craftsmanship of the person preparing the score. For instance, the pressure applied with a cutting tool or pen to “draw” stems and beams will affect their widths; the sharpness of these tools, which must be honed occasionally, will also influence the result.

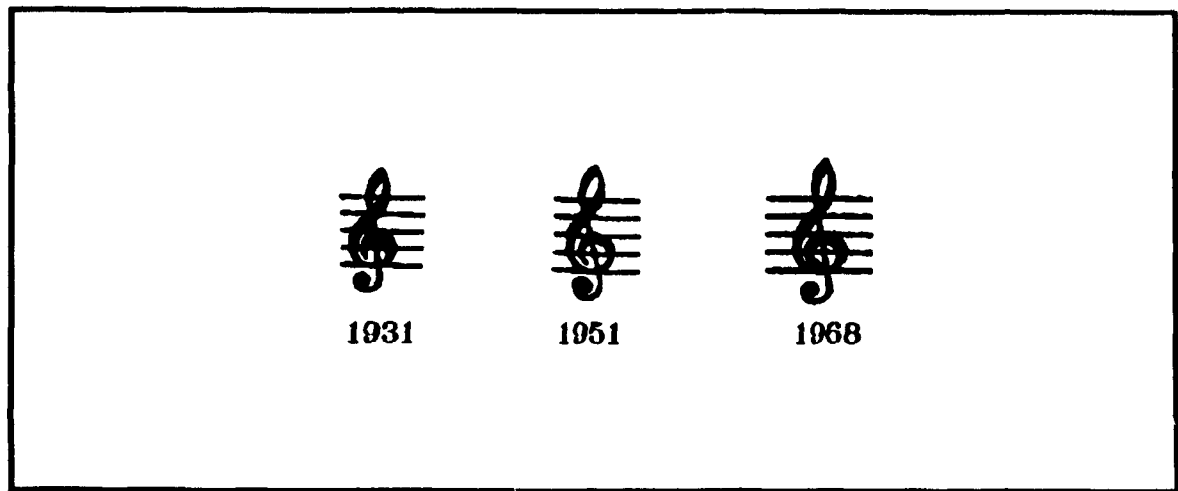


Figure 4.8. Treble clefs used by A. Leduc (Perier 1931; Brod 1951; Bozza 1968)

Finally, the inking process and the type of paper also affect the appearance of the symbols. With processes that require inking the individual “punches”, such as lithography, stamping, and stencils, variations in the amount of ink applied at each impression will alter the size and shape of the symbol. The cloth ink ribbon used in typewriters and computer printers often results in smudged types, for example, filled half-notes. Inconsistencies may also be the result of an unevenly coated inking roller on a printer, or inferior paper not absorbing the ink uniformly over its surface. Furthermore, many old editions have a problem of print-through where some impressions are visible on the reverse side, as demonstrated in figure 4.4. This figure also illustrates some symbols that are quite different from modern symbols, such as the sharps and quarter-rests.

4.2.3 Positioning

The problems of placing the symbols at their correct position are similar to those found in determining the orientation of the symbol. This is because the process is almost always performed manually and relies heavily on the ability and the experience of the individual preparing the score. Proper placement of the notehead in the vertical dimension is particularly critical. Most engraving notehead punches have a raised line on the face to facilitate the placement of the note on a line. To place the note on a space, engravers use

the staff line, which is indented, as a tactile guide to placing the punch. Such aids are unavailable to other types of punching methods. Only computers and well-aligned typewriters can ensure proper vertical placement.

Horizontal spacing of the symbols is determined solely by the person or the particular computer algorithm placing the symbols on the staff. There are no strict rules governing spacing between symbols; even when rules are given, they differ in detail. For example, Gamble states that the distance between the left side of the clef and the left side of the first note is four staffspaces (Gamble [1923] 1971, 129) (a staffspace is the distance between two staff lines), whereas Ross says this distance should be five and one-half staffspaces (Ross 1970, 145). Gamble concludes that "the eye is often the best judge for the placing of each value properly; in fact, a good many engravers use no other guide." (Gamble [1923] 1971, 132)

One standard rule useful for OMR prescribes that two symbols are not to touch. Unfortunately, this rule is not strictly followed in practice, especially in the case of a note and its accidental (see figure 4.9). Interestingly, this rule is particularly difficult for computer printing programs to comply with automatically (Byrd 1984; 165–71). Figure 4.9, although atypical, demonstrates the wide range of possibilities in positioning the symbols. Observe, for example, the flat underneath the meter signature (system 4) and the position of the ledger lines (system 7).

4.3 Conclusion

It is clear from this brief look at the potential problems facing the design of an OMR system that fairly sophisticated and flexible techniques must be developed. Simple template-matching techniques, often used in the recognition of ordinary machine-generated text, will not suffice to handle the wide variations found in the shape and orientation of the symbols encountered in musical scores. As previous research shows, staff

PARTITA

Preludio

A. ADNAN SAYGUN
Op. 36

Sostenuto (♩ = 80)

847-16

© Copyright 1964 by Southern Music Publishing Co. Inc.
 International Copyright Secured
 Printed in Italy
 All Rights Reserved including the Right of Public Performance for Profit
 "WARNING: Any person who copies or arranges all or part of the words or music of this musical composition
 shall be liable to an action for injunction, damages and profits under the United States Copyright Law."

Figure 4.9. An example of various positioning (Saygun 1964)

interference is a major obstacle to the contour-tracing method. Furthermore, contour-tracing would have difficulty handling the symbols that are attached to each other. (This problem was probably not encountered by the authors of the M.I.T. dissertations due to their limited samples.)

Because a musical score usually consists of a mixture of symbols formed by “tools” and symbols formed by hand, the level of difficulty involved in OMR lies somewhere between that of recognition of machine-generated text and of hand-written text. Although practical systems for the recognition of machine-generated English characters have been available for some time, the fact that the recognition of hand-written characters is still at an experimental stage might indicate that the design of an OMR system will be very complex. This expectation is based on the assumption that similarities exist between optical character recognition and OMR. There is, however, one trait which is unique to the set of musical symbols.

While each character in an alphabet has basically the same dimension as the other characters, most musical symbols have significantly different shapes and sizes from those of other musical symbols. It is this observation that prompted the present research to use projections as its primary recognition tool. The projection technique was expressly chosen for its inability to deal with details. This technique cannot determine the precise orientation or the exact position of a symbol. It cannot ascertain whether two symbols are touching or not. In fact, it cannot even detect that the symbols are superimposed on staves. About all it can establish is the approximate shape and size of the symbols. But this is all that is required.

That the set of musical symbols has this peculiarity, not found in regular character sets, is certainly not an accident. In reading musical symbols, there are requirements that are not imposed on reading ordinary characters, namely, precision and speed. While reading a book it is not detrimental if some letters are skipped or misread; playing a score demands that every symbol be read correctly, the first time. When reciting a written text, the pace in which it is read is usually determined by the reader; performing a piece of music requires that the score be read at the speed dictated by the music, regardless of the complexity of the

notation. During the evolution of the music notation system, these requirements were taken into consideration to ensure that music could be read accurately and efficiently.

In the next chapter, an explanation is given as to how projections and other recognition techniques are incorporated into the software to “read” music accurately and efficiently.

Chapter 5

Software Design

5.1 Overview

In this chapter, a general description of the computer program designed to recognize a small set of musical symbols on a page of music for a single, monophonic instrument is presented. The set of symbols that the software is programmed to recognize are: four types of clefs, half-notes, quarter-notes, beamed notes, flagged notes (no distinction is made between different types of flagged notes), accidentals (flat and sharp/natural), quarter-rests, eighth-rests, dots of prolongation, and barlines. The flow chart of the overall strategy is shown in figure 5.1. First, given the matrix of a digitized bi-level image, the number and location of the systems are determined; each system is then parsed from left to right to locate and identify the symbols it contains. (A system is defined here as anything that is related to a single staff, which consists of five parallel lines.) This operation is repeated until all the systems on the page have been analyzed. With one exception, all the algorithms are based on projections, since one of the major objectives of the present research is to determine the effectiveness of using projections for OMR. Thus, even when other techniques were available, projections were used if they solved a particular problem.

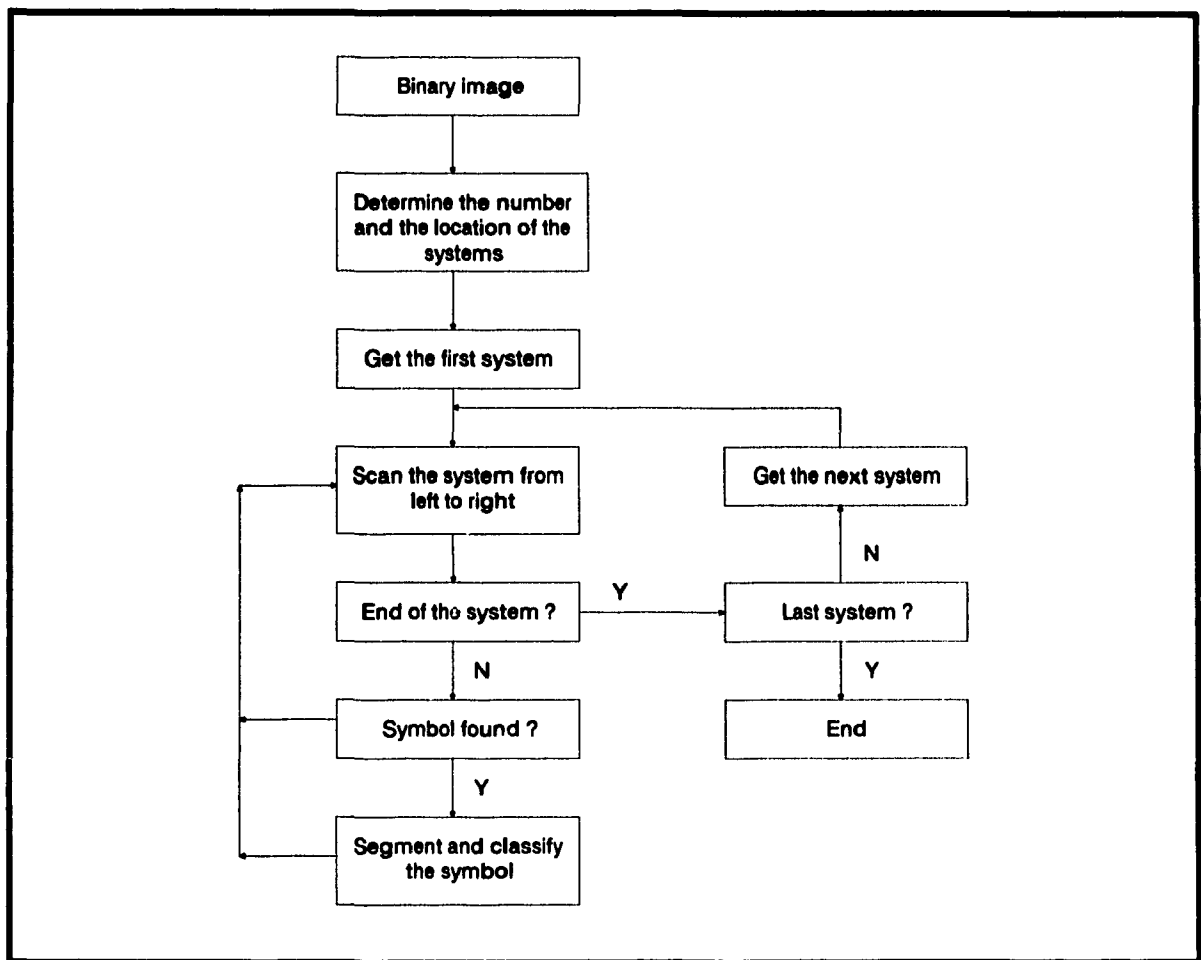


Figure 5.1. Overall Process

5.2 Locating the system

The first step is locating the systems on the page. In order to locate the systems, a y-projection of the entire page onto the vertical axis is taken (see figures 5.2–5.7). The assumption is that the staff will be represented by five peaks. Using the maximization technique described in Chapter 3 and using the mean of the entire projection as the threshold value, groups of peaks that may represent the lines of a staff are chosen. The use of the mean as the threshold value was dictated by the results of experiments with various samples of music, likewise for all other threshold values used.

The image displays a musical score for a piece titled "Mennetto secondo". The score is presented in a Y-projection format, where multiple staves are aligned vertically to show the relationship between different parts of the music. The notation includes treble clefs, a key signature of one sharp (F#), and various musical symbols such as notes, rests, and dynamic markings like *f* (forte) and *p* (piano). The score is divided into two main sections: the first section consists of six systems of staves, and the second section, labeled "Mennetto secondo", consists of four systems. The Y-projection allows for a clear view of the melodic and harmonic relationships between the different parts of the ensemble.

Figure 5.2. Y-projection and system separation (Tromlitz 1976)

PRÉLUDES
I
DEUXIÈME CAHIER

Modéré
extrêmement égal et léger
la m. g. un peu en valeur sur la m. d.

The image displays a musical score for Debussy's 'Préludes, I, Deuxième Cahier'. It features a Y-projection of the piano and organ parts. The score is written on six staves, with the piano part on the left and the organ part on the right. The tempo is marked 'Modéré' and the character is 'extrêmement égal et léger'. A performance instruction reads 'la m. g. un peu en valeur sur la m. d.' (the right hand a little more in value than the left hand). The score includes various musical notations such as notes, rests, and dynamic markings like 'pp' (pianissimo) and 'm. d. accrus' (middle hand increases). The copyright notice at the bottom reads '© Copyright 1968, by Broekmans & van Poppel, Amsterdam, Ne 101'.

© Copyright 1968, by Broekmans & van Poppel, Amsterdam, Ne 101

Figure 5.3. Y-projection and system separation (Debussy 1968)

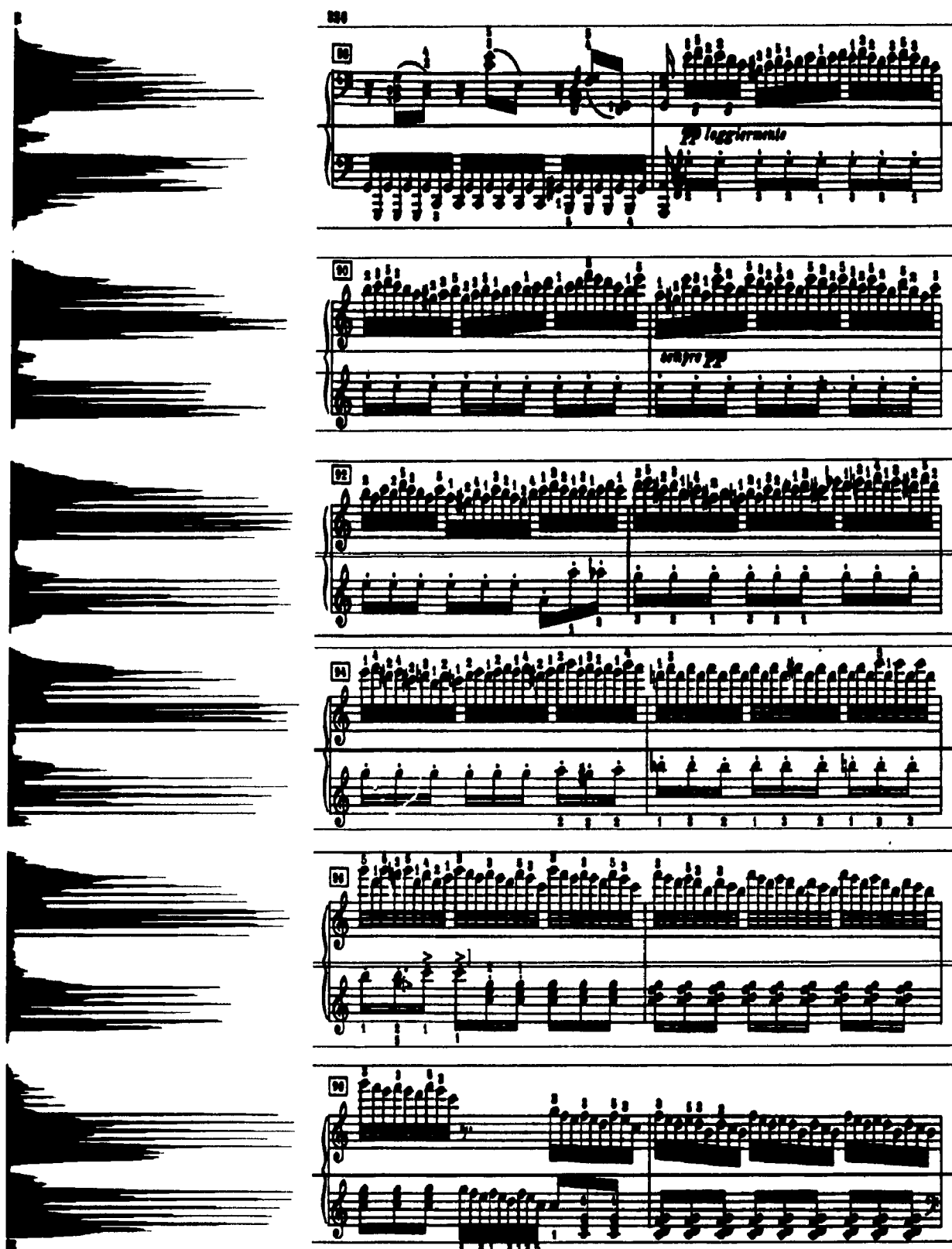


Figure 5.4. Y-projection and system separation (Beethoven 1978)

The image displays a musical score for Arnold Schoenberg's 'Verklärte Nacht' (Op. 4). The score is written for a string quartet (Violin I, Violin II, Viola, Violoncello) and a piano. The notation is dense, with many notes and rests, and includes various musical markings such as 'staccato', 'legato', and 'piano'. The score is divided into two systems, with the first system starting at measure 1 and the second system starting at measure 11. The left side of the image is heavily obscured by a large black rectangular block.

Figure 5.5. Y-projection and system separation (Schoenberg 1939)

très large

Tranquille $\text{♩} = 100$

Lent $\text{♩} = 112$ encore plus lent

en ralentissant jusqu'à la fin

EAS 17228

Figure 5.6 Y-projection and system separation (Ravel 1975)

Fließende Achtel, aber mit viel Rubato
♩ = dem vorhergehenden Triolenmaß (♩ = 72 bezeugend)

1. Vl.
2. Vl.
3. Vl.
4. Vl.
5. Vl.
6. Vl.
7. Vl.
8. Vl.
9. Vl.
10. Vl.
11. Vl.
12. Vl.
13. Vl.
14. Vl.
15. Vl.
16. Vl.
17. Vl.
18. Vl.
19. Vl.
20. Vl.
21. Vl.
22. Vl.
23. Vl.
24. Vl.
25. Vl.
26. Vl.
27. Vl.
28. Vl.
29. Vl.
30. Vl.
31. Vl.
32. Vl.
33. Vl.
34. Vl.
35. Vl.
36. Vl.
37. Vl.
38. Vl.
39. Vl.
40. Vl.
41. Vl.
42. Vl.
43. Vl.
44. Vl.
45. Vl.
46. Vl.
47. Vl.
48. Vl.
49. Vl.
50. Vl.
51. Vl.
52. Vl.
53. Vl.
54. Vl.
55. Vl.
56. Vl.
57. Vl.
58. Vl.
59. Vl.
60. Vl.
61. Vl.
62. Vl.
63. Vl.
64. Vl.
65. Vl.
66. Vl.
67. Vl.
68. Vl.
69. Vl.
70. Vl.
71. Vl.
72. Vl.
73. Vl.
74. Vl.
75. Vl.
76. Vl.
77. Vl.
78. Vl.
79. Vl.
80. Vl.
81. Vl.
82. Vl.
83. Vl.
84. Vl.
85. Vl.
86. Vl.
87. Vl.
88. Vl.
89. Vl.
90. Vl.
91. Vl.
92. Vl.
93. Vl.
94. Vl.
95. Vl.
96. Vl.
97. Vl.
98. Vl.
99. Vl.
100. Vl.

5 (letzte) Szene⁴ Vor Marias Werkstatt (Helles Morgen, Sommerzeit)
5th (last) Scene In front of Maria's house (Bright morning, Summer.)

Mariens Kinde mit einem Strohpfad rollend
Maria's child in rolling a straw-house

Kinder spielen und lachen
Children are playing and laughing

Ein gel, Ein gel, Ro. ankreum, Ein gel, rohn! Ein gel, Ein gel, Ro. ankreum.
Ein gel, Ein gel, Ro. an, Ein gel, rohn! Ein gel, Ein gel, Ro. an.

Fließende Achtel, aber mit viel Rubato
♩ = 72 bezeugend

1. Vl.
2. Vl.
3. Vl.
4. Vl.
5. Vl.
6. Vl.
7. Vl.
8. Vl.
9. Vl.
10. Vl.
11. Vl.
12. Vl.
13. Vl.
14. Vl.
15. Vl.
16. Vl.
17. Vl.
18. Vl.
19. Vl.
20. Vl.
21. Vl.
22. Vl.
23. Vl.
24. Vl.
25. Vl.
26. Vl.
27. Vl.
28. Vl.
29. Vl.
30. Vl.
31. Vl.
32. Vl.
33. Vl.
34. Vl.
35. Vl.
36. Vl.
37. Vl.
38. Vl.
39. Vl.
40. Vl.
41. Vl.
42. Vl.
43. Vl.
44. Vl.
45. Vl.
46. Vl.
47. Vl.
48. Vl.
49. Vl.
50. Vl.
51. Vl.
52. Vl.
53. Vl.
54. Vl.
55. Vl.
56. Vl.
57. Vl.
58. Vl.
59. Vl.
60. Vl.
61. Vl.
62. Vl.
63. Vl.
64. Vl.
65. Vl.
66. Vl.
67. Vl.
68. Vl.
69. Vl.
70. Vl.
71. Vl.
72. Vl.
73. Vl.
74. Vl.
75. Vl.
76. Vl.
77. Vl.
78. Vl.
79. Vl.
80. Vl.
81. Vl.
82. Vl.
83. Vl.
84. Vl.
85. Vl.
86. Vl.
87. Vl.
88. Vl.
89. Vl.
90. Vl.
91. Vl.
92. Vl.
93. Vl.
94. Vl.
95. Vl.
96. Vl.
97. Vl.
98. Vl.
99. Vl.
100. Vl.

⁴ Das Orchester dieser Szene hat auf folgende Weise Besetzung reduziert: 4 pr. Vl., 1 Ob., 1 Kl. in Es, 2 Kl. in B, 1 Bcl., 4 Hörner, 1 Trp., 1 Tuba, das gesamte Schlagwerk, Xyl., Cel., Hrn. und die Streicher ohne Kb.

Figure 5.7. Y-projection and system separation (Berg 1955)

The distance between peaks is calculated next. A cluster of five or more peaks that is separated from other clusters of peaks by a certain distance is designated as the minimum boundary of a system. (The separation of systems in figures 5.2 to 5.7 is indicated by thin horizontal lines drawn by the program.) The reason for including more than five peaks is to embrace peaks generated by long horizontal beams (see figure 5.3) and ledger lines (see figure 5.4). Note that this method of separating the systems does not make any presumptions about the height of each staff, therefore it successfully segments music with different sizes of staves on a page (as shown in figure 5.6). On the other hand, it does assume that the staff has at least five lines, therefore the one-line staff sometimes used for percussion instruments is overlooked (see figure 5.7). This minor defect excepted, the method is extremely reliable since it makes very few assumptions. The method will work even if the stafflines are not perfectly parallel to each other or to the top edge of the page; in fact, the lines need not even be equidistant from each other. It also allows for some of the lines to be relatively faint, or broken, as is often the case in typeset music.

The top and bottom boundaries of a system are identified either by searching for the first blank line (very low projection value) above and below the minimum boundary, or, in the absence of a blank line, by using the minimum projection value between the neighbouring systems. The lack of a blank line may be due to noisy input, but in most cases it is due to overlapping symbols from two systems. This overlapping is often found in dense orchestral scores (figure 5.5), in music for keyboard instruments, and in other settings where one instrument or a group of voices uses more than one staff (figures 5.3 and 5.4). In the latter case, the partitioning problem can be solved by specifying, beforehand, the number of staves to be considered together. The separation of two one-staff systems that overlap, however, is a complex problem and remains unsolved. This problem was not encountered in samples of music for a single, monophonic instrument, which is the primary target of this project. The remainder of the recognition process applies to this type of music only.

5.3 Analysis of a system

After the separation process, each system is analyzed independently and sequentially to locate and classify the symbols it contains. (Given a suitable environment, each system could be processed in parallel.)

5.3.1 Locating the staff

The first step is to determine the exact location of the staff within the given system. Series of y-projections are taken from a tall and narrow rectangle area, which moves inward starting at the right margin, until five clear peaks appear. The width of the rectangle in the current implementation is 0.254cm (0.1inch) and its height is the same as that of the system. The reason for starting at the end of the staff is that it is rare to find anything to the right of the staff, while there may be spurious symbols to the left of the staff, such as the name of the instrument. The routine for finding the beginning of the staff is similar, but it uses as an aid the height of the staff, which was obtained when the end of the staff was located. It should be noted that, in general, the vertical position of the staff is not the same at its left and right sides because the staff is skewed; see figure 5.2 for an example. Since some of the subsequent recognition algorithms are sensitive to the position of the staff, the exact vertical position is updated periodically as the staff is scanned from left to right. The height of the staff is also used to derive the staffspace, which is the distance between two adjacent staff lines. It is assumed that the size of musical symbols is linearly related to the staffspace, thus this value is used as the normalization constant. The assumption that a linear relationship holds between the size of the symbols and the size of the staffspace is not rigorously examined. Informal observations reveal, however, that although there is a wide variance in the size of the symbols within a given staff size, the size of the symbols does tend to increase linearly as the staff size increases. This assumption was also made by Prerau in his experiment.

5.3.2 Locating the symbols

The next step is to take the x-projection of the entire staff in order to locate the individual musical symbols. Originally, a projection of the entire height of the system was taken, but it was found that interference from such things as expression markings and measure numbers made recognition difficult (see the first and



Figure 5.8. X-projection of the entire system



Figure 5.9. X-projection of the staff only

the fifth measure in figure 5.8). Therefore, the rectangle used for the x-projection is defined by the staff and not by the system, as shown in figure 5.9. This projection, used mainly to locate the symbols, will be referred to as the staff projection. Although this modification solves the problem of interference, this represents only a partial projection of the symbols unless they lie entirely on the staff. For example, see the second beam group of measure 3 in figure 5.9. (The solution to this problem will be given below.) Before the staff projection can be used to locate the symbols, it is scanned for the global minimum, which is assumed to be the total contribution from the staff lines. This value will be considered as noise while searching for symbols in the staff projection.

In general, the process of locating and segmenting a symbol works as follows. The staff projection is scanned from left to right; where the projection value is greater than the noise plus one staffspace, it is considered a projection contributed by part of a symbol (see figure 5.10 for an illustration). At this point, a local y-projection is taken, using the rectangle delimited at the top and bottom by the height of the entire system, at the left side by $(x - \text{staffspace})$, and at the right side by $(x + 3 * \text{staffspace})$, where x is the position located

in the scanning process above (the tall rectangle in figure 5.10). This projection is used to determine if the symbol extends above and below the staff and thus compensates for the limitation of the staff projection, which does not include the entire symbol.

Once the vertical boundaries of the symbol are found, a local x-projection is taken, using the same rectangle but with the new vertical limits (the embedded rectangle in figure 5.10). Within this area, the projection contributed by the staff lines must be recalculated since this value tends to vary across the page. The projection of the symbol alone can now be determined by subtracting the projection of the staff lines from the staff projection. With this final projection, it is simple to calculate the width, the maximum height, and the number of vertical peaks using the maximization routine; the area (the total projection values) of the symbol is determined by the resulting projection profile.

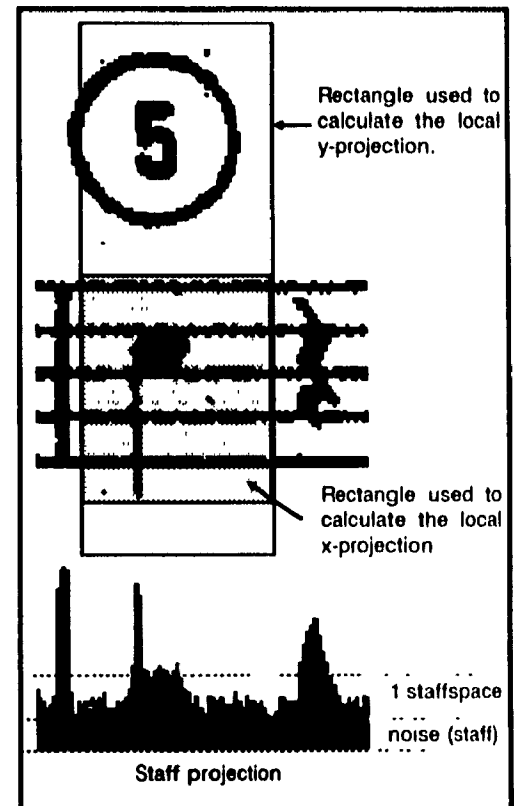


Figure 5.10. Locating the symbols

5.3.3 Clef classification

The first symbol the program expects while parsing the staff projection is the clef. Currently, the treble clef, the alto clef, the tenor clef, and the bass clef are defined. Any other clef or symbol occupying the left-most area of the staff will be mistaken for one of the four clefs, or an error message will appear indicating an unrecognizable symbol. After the first symbol is segmented, the classification scheme uses the maximum height, the width, and the area of the symbol, as reflected in the projection (see figure 5.11). In some cases,

where these features are insufficient to distinguish between a treble and a bass clef, a y-projection is taken between the fourth and the fifth staff lines. This is very effective since most bass clefs do not occupy this area.

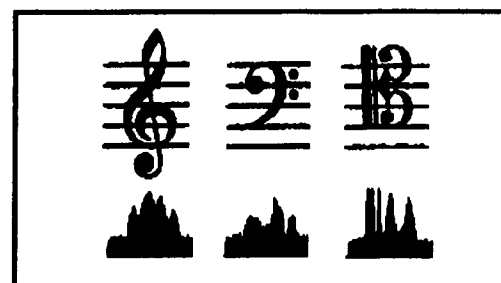


Figure 5.11. Projections of clefs

5.3.4 Key Signature Classification

To find the type (sharp or flat) and the number of accidentals in the key signature, the staff projection is scanned from the right side of the clef until an empty space larger than a staffspace is found. The assumptions here are that the space between the accidentals within a key signature is less than a staffspace, and that the space between the last accidental in the key signature and the following symbol is at least a staffspace. When peaks are found between the right side of the clef and the location of the empty space, the number of peaks in the region thus defined is calculated with the maximization routine. Since a key signature involving sharps will have twice the number of peaks than one involving flats, it is possible to deduce whether the key signature comprises flats or sharps, given the total width of the region and the predetermined minimum width of accidentals. Assuming that the position in which accidentals appear in key signatures is fixed, no further processing is needed. No suitable projection-based algorithm was found to recognize the meter signatures because of the many different shapes of numerals found in the scores. Consequently, the use of meter as part of the recognition strategy, described in Chapter 3, is not implemented in the current software. Some type of template-matching method may solve the meter signature problem.

5.3.5 Classification of other symbols

Classification of the other symbols involves calculating their width and height by the process described in 5.3.2. Using only these two features, the symbols can be subdivided into eight classes:

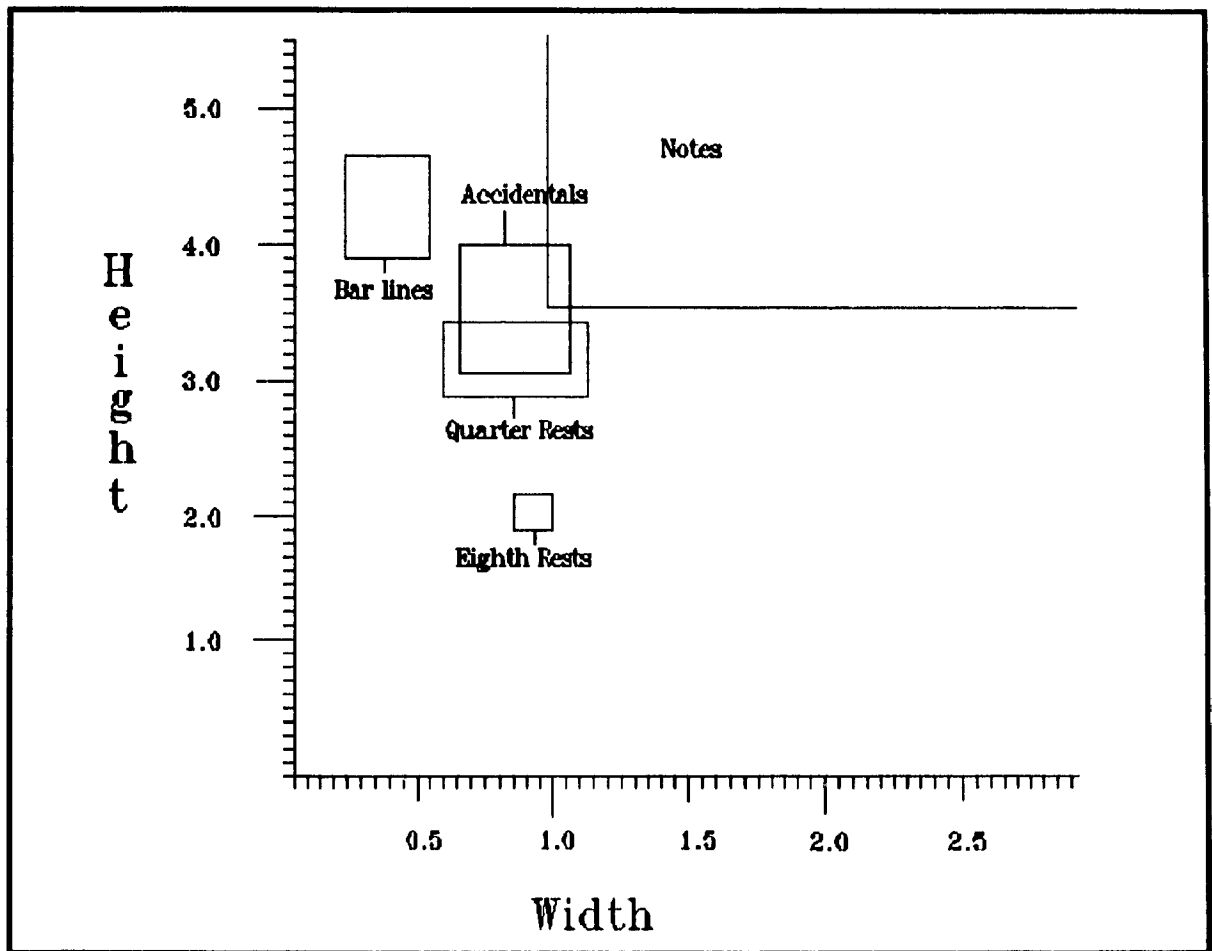


Fig. 5.12. Height-Width Plane

- Class 1. Not a symbol
- Class 2. Barline
- Class 3. Quarter-rest
- Class 4. Eighth-rest
- Class 5. Accidental
- Class 6. Accidental or Quarter-Rest
- Class 7. Note
- Class 8. Unknown symbol

Why such a classification is possible can be understood from figure 5.12, where the rectangular regions occupied by each of these symbols are shown in the normalized height-width plane (scaled to one staffspace). Each rectangle represents the region in which the symbol or the group of symbols may be found according to its height and width. The size of the rectangle for each symbol was determined by surveying many scores

from various publishers. For example, it was found that the dimensions of most eighth-rests are approximately two staffspaces in height and one staffspace in width. (A similar method of classification was also used by Prerau.) Note that, if the rectangle for a single symbol does not intersect with any other rectangles, the classification of this symbol is trivial. If, on the other hand, the rectangles overlap, further analysis is needed to differentiate the symbols.

If the width of a symbol is less than that of a barline, it is ignored, and the program resumes the scanning of the staff projection. If a rest is found (as in Class 3 and 4), the only thing left to do is to see if there is a dot attached to it. The method of locating the dot will be described later. If the number of vertical peaks of an accidental (Class 5) is one it is a flat, otherwise it is classified as a sharp or a natural. No reliable technique using projections was found to distinguish between sharps and naturals. Within Class 6, if the number of vertical peaks is two, the symbol is either a sharp or a natural. If there is only one peak, the position of the peak with respect to the entire width of the symbol is determined to decide whether it is a flat or a quarter-rest. If the peak is located in the left quarter of the symbol width it is recognized as a flat; if not, it is considered a quarter-rest (see figure 5.13).

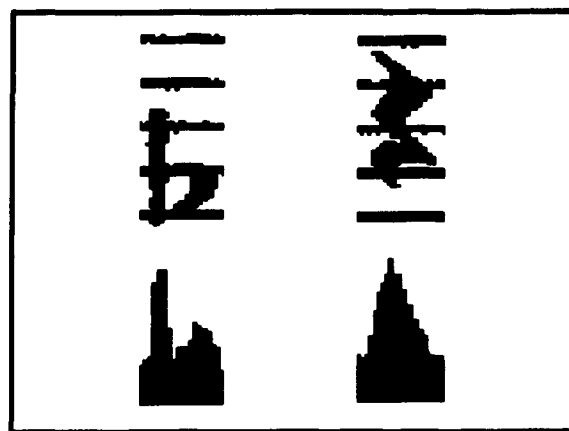


Figure 5.13. Projections of flat & quarter-rest

If the symbol is in Class 7 it may be a quarter-note, a half-note, a flagged note, or a beamed note. For this group another feature is extracted to simplify the classification process: a y-projection on each side of the stem is calculated to determine the number of horizontal peaks, ignoring the staff lines. These horizontal peaks are used to establish the presence of the notehead, flags, and beams. As shown in figure 5.14, the number of horizontal peaks on the right and left sides of the beam (abbreviated as R_Peak and L_Peak) is used along with other information to further classify the symbols of Class 7. Although flags and beams are

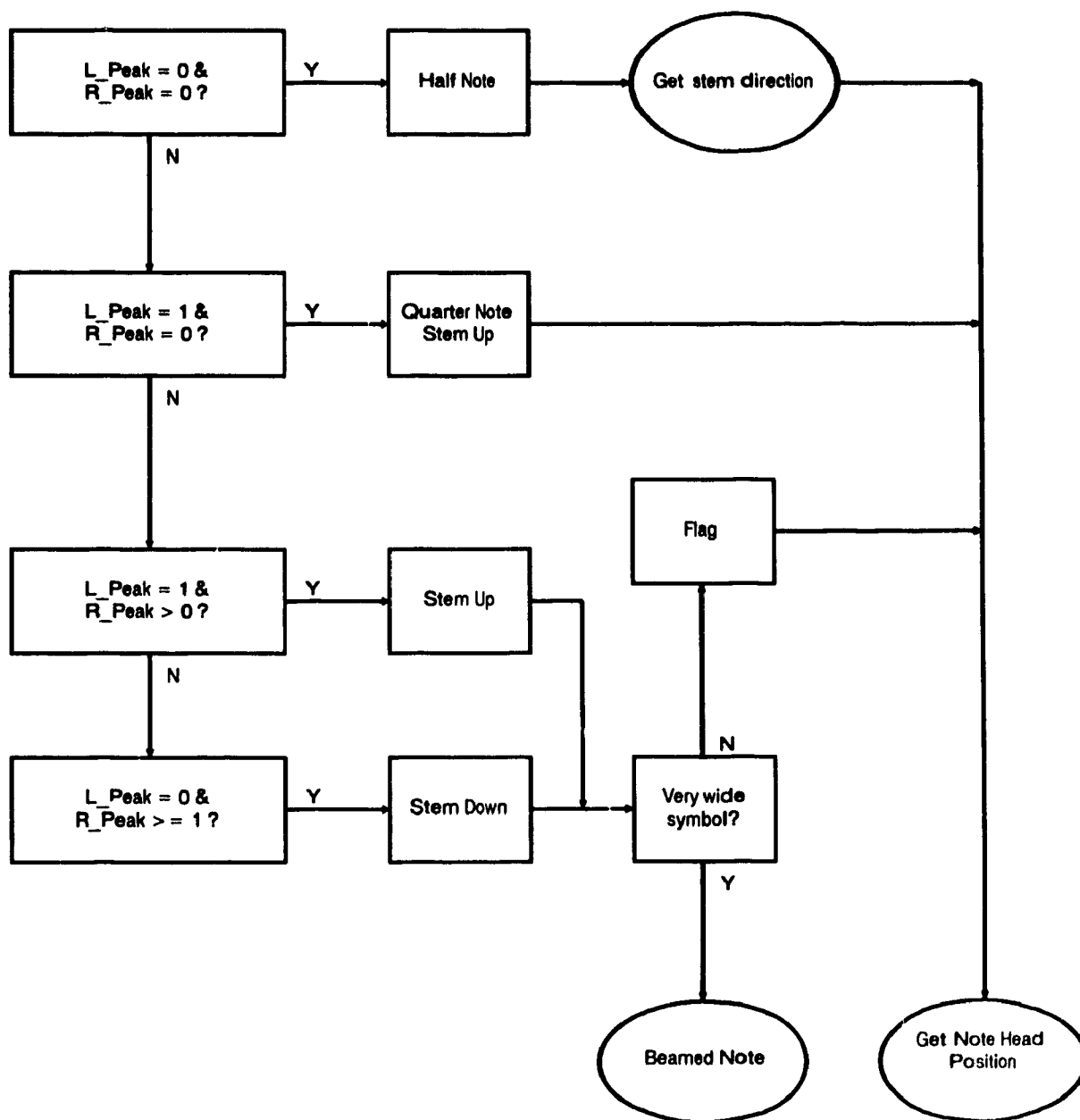


Figure 5.14. Note Classifier

always evident as horizontal peaks, the notcheads of half-notes are unpredictable, sometimes appearing as peaks, other times not. Thus, in the absence of flags or beams, the distinction between quarter-notes and half-notes is not made until after the position of the notehead becomes available. Where no peaks are present, the direction of the stem is determined by the location of the maximum peak (the stem) in the local x-projection, with respect to the entire width of the symbol.

5.3.6 Beamed notes

If the symbol is found to have beams attached to it, a modified recognition strategy is used to process the symbols within the beamed group. The beamed note groups are treated differently for two reasons. First, because the staff projection used to locate the symbols excludes anything outside the staff, a note attached to the beams and lying entirely above or below the staff will be missed in the regular symbol-locating process. Second, the classification scheme can be made much simpler for beamed notes than for symbols standing alone. This is because, syntactically, the number of different symbols that can appear in the beamed region is significantly reduced. For example, half-notes and flagged notes cannot be encountered. In the current program, the only symbols expected are notes and accidentals; the rests allowed in some notational practices as part of beamed groups are not recognized.

Instead of using the staff projection, an alternate x-projection is taken where the top and bottom boundaries of the previous note in the beamed group are used as vertical boundaries. This ensures that no beamed symbols are bypassed. The rest of the recognition process is similar to that of the isolated symbols except that the classification process is modified because of the reduced number of possible targets, and because the horizontal left and right peaks are now used to count the number of beams attached to the left and right sides of the stem. When the number of beams on the right side is zero, this signals the end of the beamed note group and the program reverts to normal processing.

53.7 Determining the notehead position

The position of the notehead is determined by taking a local y-projection either to the right or to the left of the stem, depending on the stem direction (which is already known); then the position of the maximum area on the projection profile is accepted as the location of the notehead. If the type of notehead, whether filled or not filled, is not known by this stage, the area calculated above is used to determine the type. If the area is above a certain threshold value it represents a filled notehead; if it is below another threshold value it represents a non-filled notehead. In the case where the area lies between the two threshold values, the ratio of the number of black pixels over the number of white pixels in a small rectangle positioned near the centre of the notehead is used to distinguish between the two types. This is the only place in the computer program where projections are not directly involved in the recognition process.

53.8 Locating the dot

Finally, if the symbol found is a note or a rest, check is made to see if there is a dot of prolongation affixed. A y-projection is taken from a small rectangle beside the symbol, where the dot may appear. If the projection profile contains a small bump, it is considered a dot. There are two reasons for detecting the presence of the dot in this manner; first, any small symbol such as a dot can be easily buried in the “noise” of the staff projection, thus searching for it while scanning the projection is highly unreliable; and second, the notational rule dictates that the only place where a dot of prolongation may appear is to the immediate right of a note or rest, hence it is futile to look for dots anywhere else.

At this point, the program will look for the next symbol in the staff projection; when a possible candidate is found, the whole process is repeated until the end of the staff is reached.

Chapter 6

Experiment and conclusions

In this final chapter, the actual implementation of the program described in the previous chapter and the result of an experiment to test the program are described. Possible improvements to the program are considered in the concluding section.

6.1 Hardware and software

The optical scanner used for the development was a Datacopy 710 which has a resolution of 78.74 dots per cm (200 dots per inch). Most current desktop scanners work as follows: the percentage of light reflected by the illuminated image is measured as a voltage level using a CCD (charged coupled device), then the analog voltage levels are converted into digital bit patterns by an analog-to-digital converter. The program was developed using the Microsoft C compiler versions 4.0 and 5.0 on various IBM-PC-type microcomputers.

6.2 Music samples and development

The software was developed in a series of trials using samples of music from various publishers. (The complete list of music used is given in Appendix A.) Many different algorithms and threshold values were tried until a satisfactory recognition rate was achieved with the training samples. The complete set of samples was used for testing the system's segmentation process. 17 samples are monophonic, these were used for the remaining stages of the recognition process.

The basic strategy for building this program was trial and error. Due to the complexity and the variety of the data, it was the only way of verifying the validity of a particular algorithm or threshold value. At each trial, modifications were made to decrease the rate of misrecognition. If the modification did correct the error, other samples were used to test the generality of the modification and to check for side-effects (that is, to ensure that the modification did not degrade the recognition of other symbols).

The most frustrating aspect of developing this system was the difficulty of monitoring progress. Because there are several steps involved before any decision is made about the symbols, it was extremely hard to locate problem areas. It was particularly difficult to determine whether misrecognitions occurred because of segmentation errors or because of classification errors.

Given that a large body of literature exists to explain various classification methods, much of the effort was devoted to developing a reliable technique to segment the symbols. The premise was that, as long as the symbols are segmented correctly, more sophisticated classification methods can be implemented later if the current one turns out to be inadequate. If, on the other hand, the segmentation is unreliable, recognition will fail no matter how sophisticated the classification scheme is.

Observing how the program is managing the data is complicated by the graphic nature of the data. Simple print statements, normally used to monitor the progress of a conventional program, were not very in-

formative. To facilitate monitoring, various subprograms were written to show graphically the state of the program, including subroutines to plot the x- and y-projections.

As development progressed it became evident that it is relatively easy, albeit time-consuming, to fine-tune the program to achieve a near-100% recognition rate with the training samples (for example, the program can recognize all of the required symbols in figure 6.1); therefore, the final test was conducted using four carefully chosen samples that were not part of the development samples. Three of the four samples were chosen to establish the worst-case performance of the program. The first two contain symbols that are quite different in shape from those found in the development samples. The third is a hand-written score, and the fourth was chosen to represent a typical score. The selection was made after the development of the program and no changes were made to the program once the samples were chosen. Absolutely no information regarding the score is supplied at execution time. In other words, the process is completely automatic except for placing the score on the scanner.

6.3 Results

With all four samples, system separation was completely successful (see Figures 6.2 – 6.5). The summary of the results of the tests is shown in Figure 6.6 (see also figure 6.7 for partial output of the program for each of the test samples). The recognition rate is calculated as the total number of correctly identified stock symbols (the set of symbols that the program is designed to identify) divided by the total number of stock symbols on the page. A note is considered correctly identified if and only if both the pitch and the duration are correct. The scanner takes about 30 seconds to read a page, and, on a IBM-AT-compatible microcomputer, the average processing time is approximately 15 seconds per system.

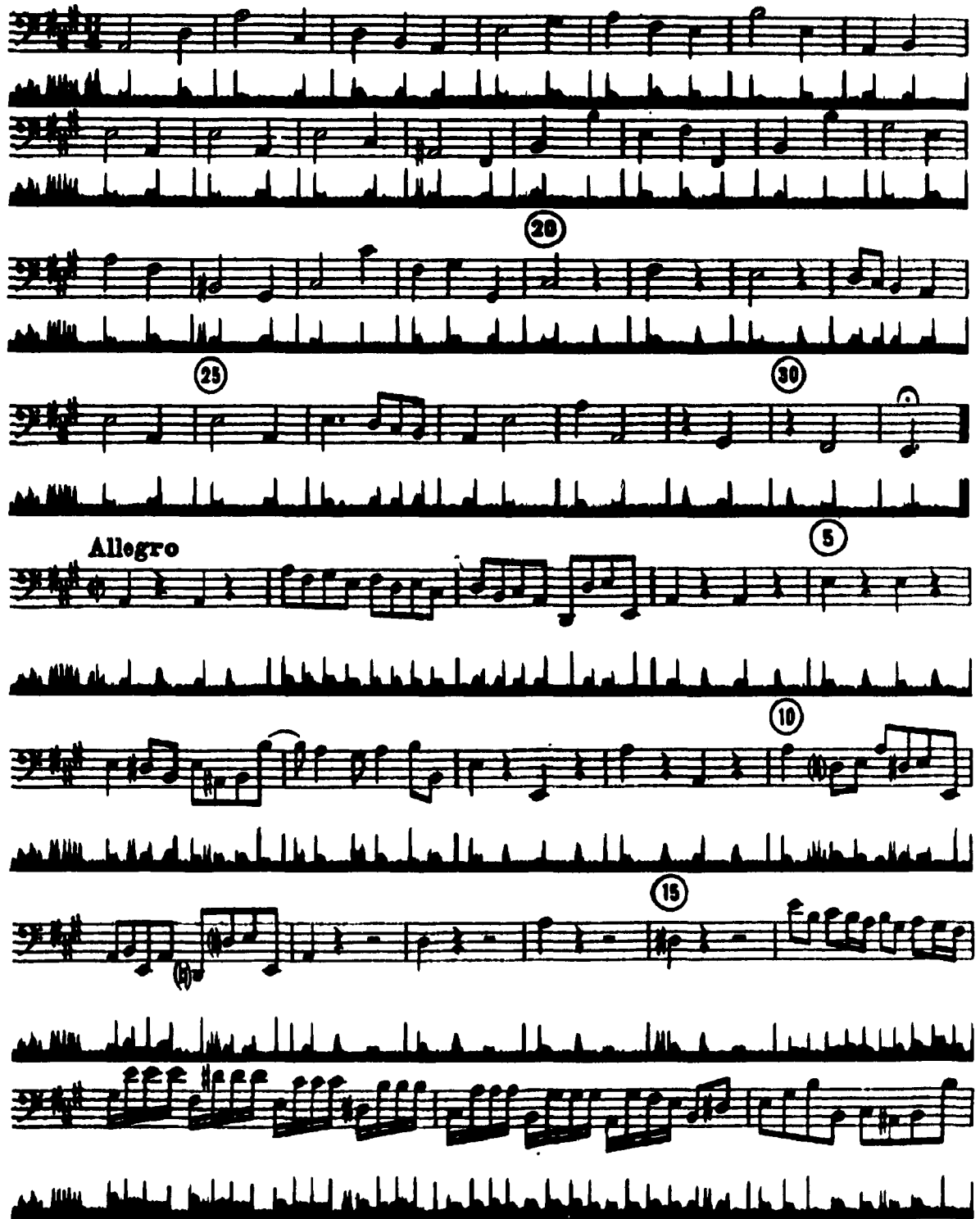
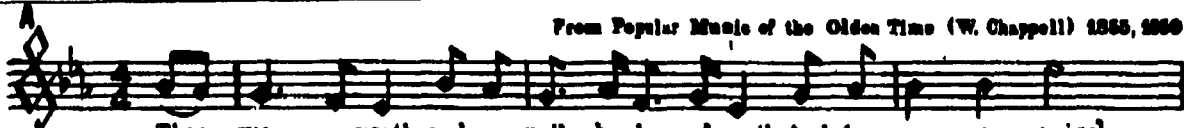



Figure 6.1. A development sample with staff projections (Teleman 1969)

The Bayliff's Daughter of Islington

A From Popular Music of the Olden Time (W. Chappell) 1865, 1880

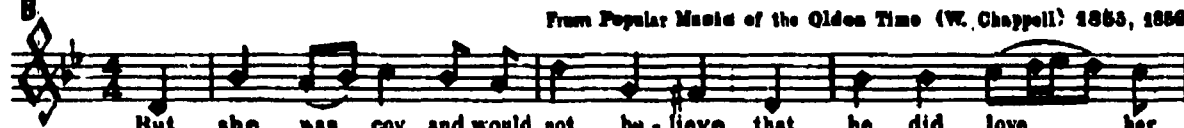


There was a youth, and a well - be - lov - ed youth, And he was a squire's




son, He — loved the bay - liff's daught - er dear That lived in — Is - ling - ton.

B From Popular Music of the Olden Time (W. Chappell) 1865, 1880



But she was coy and would not be - lieve that he did love — her



so. No, not at an - y — time she would An - y coun - ten - ance to him show.

Figure 6.2. Sample I (Goss 1937)

LUCIEN PATÉ
ADAPTED

Roses in Autumn

CÉSAR FRANCK

Slowly

1. The rose that bloomed in sum - mer gar - dens, fra - grant and red,
2. But deep with - in the roots the ros - es sleep till the spring

Is faded now, its pet - als ly - ing scat - tered and dead.
When birds re - turn and in the gar - den joy - ful - ly sing,

And all the birds of sum - mer-time are van - ished and fled
Till all the air is sweet with songs that ech - o and ring

And leaves are drift - ing from the branch - es o - ver my head.
With mag - ic sound from leaf - less stalk, the ros - es to bring.

ST. THOMAS AQUINAS

Panis Angelicus

CÉSAR FRANCK

12

Pa - nis an - gé - li - cus fit pa - nis hó - mi - num;

Dat pa - nis cée - li - cus fi - gú - ris tér - mi - num;

O res mi - rá - bi - lis! man - dú - cat Dó - mi - num

Pau - per, pau - per, sér - vus et hú - mi - lis,

Pau - per, pau - per, sér - vus et hú - mi - lis.

110

Figure 6.3. Sample II (Cecilia 1960)

35. SWING LOW SWEET CHARIOT

Briskly - with no variation in tempo R. Mac Gimsey

Oh — swing low, — Oh swing low, — Oh

swing low sweet Char-i - ot — Swing low.

It mus' be Je - sus Pass-in' by — Oh

swing low sweet Char-i - ot — Swing low. Swing low in nuh

Eas' swing low swing low in nuh Eas' swing low, swing low char-

- i - ot swing low, — swing low — char - i - ot swing low.

Figure 6.4. Sample III (Parrish 1942)

STREICHQUARTETT

Basso **A-dur** **Opus 9 Nr. 6**
Hoboken III: 24

Presto

The musical score for the Bassoon part of Haydn's String Quartet Opus 9 No. 6, Hoboken III: 24, is presented in A major, 3/4 time, and marked Presto. The score consists of eight staves of music. The first staff begins with a forte (f) dynamic and a piano (p) dynamic. The second staff features a forte (f) dynamic. The third staff also features a forte (f) dynamic. The fourth staff features a forte (f) dynamic. The fifth staff features a forte (f) dynamic. The sixth staff features a forte (f) dynamic. The seventh staff features a piano (p) dynamic. The eighth staff features a piano (p) dynamic.

Figure 6.5. Sample IV (Haydn 1968)

	Number of symbols correctly identified	Total number of symbols	Percentage of correct symbols
Sample I	79	111	71%
Sample II	138	211	64%
Sample III	92	140	66%
Sample IV	281	344	82%
Total	590	806	73%

Figure 6.6. The result of the experiment

Sample I — This page was chosen for its idiosyncratic collection of musical symbols: the bizarre treble clef, the diamond-shaped noteheads, and a form of flags normally found in music set with types, although the inaccuracy in the placement of noteheads indicates that it was not typeset. This sample also violates the rule that states that every system starts with a clef. Despite these obstacles, the program performed relatively well, correctly identifying 71% of the symbols including the two treble clefs. Obviously, the program was confused at the beginning of the second and fourth system, not being able to find the expected clef. Other misrecognitions include the first quarter note in the first system, third bar (A-flat instead of B-flat) and the two quarter notes in the third system, measure three (stems too short). Nevertheless, its ability to deal with the diamond-shaped noteheads and the short flags demonstrates the flexibility of the projection technique.

Sample II — The relatively poor recognition rate of this sample results largely from the odd shape of some of the music symbols used. Both the eighth- and quarter-rests are wider than usual and the treble clefs have thinner strokes than those encountered during program development. As a side effect, the program also had difficulty locating the sharp in the key signature. If these symbols are disregarded, the recognition rate is increased to 80%. The problems encountered in this sample are mainly due to limitations in the classification stage, which basically cannot deal with any symbol that it has not seen before. This can be remedied by using a more sophisticated classifier, possibly with a database.

Sample I

TREBLE CLEF
3 FLAT(S)
FLAG_UP 1 SPACE 4
DOT
BEAM_UP LINE 3 BEAM 1
BEAM 1 SPACE 3 BEAM 0
BAR LINE
UP BLACK LINE 4
DOT
FLAG_UP 1 SPACE 4
UP LINE 5
FLAG_UP 1 LINE 3
FLAG_UP 1 SPACE 3
BAR LINE
FLAG_UP 1 SPACE 4
FLAG_UP 1 SPACE 3
FLAG_UP 1 SPACE 4
FLAG_UP 1 LINE 4
UP SPACE 5
FLAG_UP LINE 4
FLAG_UP SPACE 3
BAR LINE
DN BLACK SPACE 3
DN BLACK LINE 3
DN WHITE SPACE 1
BAR LINE

UNKNOWN CLEF
1 FLAT(S)
UP WHITE SPACE 4
BAR LINE
UNKNOWN SYMBOL
BEAM DN 1 SPACE 2 BEAM 1
BEAM 2 SPACE 2 BEAM 1
BEAM 1 LINE 2 BEAM 0
BLACK SPACE 1
BEAM DN LINE 2 BEAM 1
BEAM 1 SPACE 2 BEAM 0
BAR LINE
UNKNOWN SYMBOL
FLAG_UP 1 SPACE 3
UP BLACK LINE 4
UP BLACK LINE 5
BAR LINE
BEAM_UP LINE 4 BEAM 1
BEAM 1 SPACE 4 BEAM 0
BEAM_UP LINE 5 BEAM 1
BEAM 1 SPACE 4 BEAM 0
UP BLACK LINE 5
UP BLACK SPACE 5
BAR LINE
UP WHITE LINE 5
UNKNOWN SYMBOL

TREBLE CLEF
2 FLAT(S)
FLAG_UP 1 SPACE 4
BAR LINE
BAR LINE
UP BLACK LINE 3
BEAM_UP SPACE 3 BEAM 1
BEAM 1 LINE 3 BEAM 0
DN BLACK SPACE 2
FLAG_UP 1 LINE 3
FLAG_UP 1 SPACE 3
BAR LINE
DN BLACK LINE 2
UP BLACK LINE 4
UP WHITE SPACE 4
UP BLACK SPACE 4
UP BLACK LINE 5
BAR LINE
SHARP/NAT
SHARP/NAT
BEAM DN SPACE 2 BEAM 1
BEAM 1 LINE 2 BEAM 2
BEAM 2 SPACE 1 BEAM 1
BEAM 1 LINE 2 BEAM 0
SHARP/NAT
UNKNOWN SYMBOL
BAR LINE
TENOR CLEF

Sample II

BASS CLEF
2 FLAT(S)
FLAG_UP 1 LINE 5
BAR LINE
UP BLACK LINE 4
SHARP/NAT
UP BLACK SPACE 3
UP BLACK LINE 3
FLAG_UP 1 LINE 5
BAR LINE
UP BLACK LINE 4
SHARP/NAT
FLAG_UP 1 SPACE 3
UP BLACK LINE 3
UP BLACK LINE 5
BAR LINE
BLACK SPACE 4
UNKNOWN SYMBOL
UP BLACK SPACE 3
FLAG_UP 1 SPACE 3
BAR LINE
UP BLACK LINE 4
UNKNOWN SYMBOL
UNKNOWN SYMBOL

BASS CLEF
FLAG_UP 1 LINE 4
BAR LINE
DN BLACK LINE 3
SHARP/NAT
DN BLACK SPACE 2
DN WHITE SPACE 1
FLAG_UP 1 LINE 4
BAR LINE
DN BLACK LINE 3
SHARP/NAT
UNKNOWN SYMBOL
DN BLACK LINE 2
FLAG_UP 1 LINE 4
BAR LINE
UNKNOWN SYMBOL
DN WHITE SPACE 2
DOT
DN WHITE SPACE 2
FLAG DN 1 SPACE 2
BAR LINE
DN WHITE SPACE 2
UNKNOWN SYMBOL
UNKNOWN SYMBOL

BASS CLEF
FLAG_UP 1 SPACE 3
BAR LINE
UP BLACK LINE 3
FLAG_UP 1 SPACE 4
UP BLACK SPACE 4
FLAG_UP 1 LINE 3
BAR LINE
SHARP/NAT
BEAM_UP LINE 4 BEAM 2
BEAM 2 SPACE 4 BEAM 0
UP BLACK LINE 5
BEAM_UP LINE 5 BEAM 1
BEAM 1 SPACE 1 BEAM 0
UP BLACK LINE 4
DOT
UP BLACK LINE 5
FLAG_UP 1 LINE 5
BAR LINE
UP BLACK LINE 4
DOT
UNKNOWN SYMBOL

BASS CLEF
1 SHARP(S)
DN WHITE SPACE 2
BAR LINE
DN BLACK SPACE 1
DN BLACK LINE 3
DN WHITE SPACE 2
DN BLACK SPACE 1

Sample III

TREBLE CLEF
2 SHARP(S)
UNKNOWN SYMBOL
UNKNOWN SYMBOL
BEAM_UP SPACE 5 BEAM 1
BEAM 1 SPACE 4 BEAM 0
BAR LINE
FLAG_UP 1 SPACE 3
UP BLACK SPACE 3
DOT
DN BLACK SPACE 2
UP BLACK SPACE 5
BAR LINE
FLAG_UP 1 SPACE 3
UP BLACK SPACE 4
FLAG_UP 1 SPACE 5
FLAG_UP 1 SPACE 6
UNKNOWN SYMBOL
UNKNOWN SYMBOL
UP BLACK SPACE 5
BAR LINE

TREBLE CLEF
4 FLAT(S)
UNKNOWN SYMBOL
FLAG_UP 1 SPACE 3
UP BLACK LINE 5
FLAG_UP 1 SPACE 4
FLAG_UP 1 SPACE 5
DOT
FLAG_UP 2 SPACE 5
BEAM_UP SPACE 6 BEAM 2
DOT
BEAM 2 LINE 7 BEAM 1
BEAM 1 SPACE 6 BEAM 0
BAR LINE
FLAG_UP 1 SPACE 5
UP BLACK SPACE 5
DOT
UNKNOWN SYMBOL

TREBLE CLEF
4 FLAT(S)
UNKNOWN SYMBOL
UNKNOWN SYMBOL
UNKNOWN SYMBOL
FLAG_UP 1 SPACE 6
DOT
BAR LINE
FLAG_UP 1 SPACE 5
UP BLACK SPACE 5
FLAG_UP 1 SPACE 4
BEAM_UP SPACE 4 BEAM 2
BEAM 2 LINE 5 BEAM 1
BEAM 1 SPACE 5 BEAM 0
UP BLACK SPACE 5
BAR LINE
FLAG_UP 1 SPACE 4
UP BLACK SPACE 4
FLAG_UP 1 SPACE 4
UP BLACK SPACE 5
UP BLACK SPACE 5
BAR LINE

TREBLE CLEF
2 SHARP(S)
UNKNOWN SYMBOL
FLAG_UP 1 SPACE 3
UP BLACK LINE 5
FLAG_UP 1 SPACE 4
FLAG_UP 1 SPACE 5
DOT
FLAG_UP 2 SPACE 5
BEAM_UP SPACE 6 BEAM 2
BEAM 3 LINE 7 BEAM 1
BEAM 1 SPACE 6 BEAM 0
BAR LINE
FLAG_UP 1 SPACE 5
UP BLACK SPACE 5
DOT
UNKNOWN SYMBOL
DN BLACK LINE 2

Sample IV

BASS CLEF
3 SHARP(S)
UP WHITE SPACE 4
EIGHTH REST
BAR LINE
EIGHTH REST
EIGHTH REST
FLAG DN 1 LINE 1
DN BLACK SPACE 1
DN BLACK SPACE 0
BAR LINE
DN BLACK LINE 1
FLAG DN 1 LINE 1
DN BLACK SPACE 1
DN BLACK SPACE 0
BAR LINE
DN BLACK LINE 1
EIGHTH REST
Q REST
EIGHTH REST
BAR LINE
DN BLACK LINE 3
SHARP/NAT
BAR LINE
DN BLACK LINE 2
DOT
DN BLACK SPACE 2
DOT
BAR LINE
DN BLACK LINE 3
SHARP/NAT
DN BLACK LINE 3
DN BLACK SPACE 2
DOT
BAR LINE

BASS CLEF
3 SHARP(S)
BEAM_UP SPACE 4 BEAM 1
BEAM 1 SPACE 4 BEAM 1
BEAM 1 SPACE 4 BEAM 0
UP BLACK SPACE 4
EIGHTH REST
EIGHTH REST
BAR LINE
FLAG_UP 1 BLACK SPACE 4
UP BLACK LINE 5
FLAG_UP 1 LINE 4
DOT
BAR LINE
UP BLACK SPACE 4
FLAG_UP 1 SPACE 4
UP BLACK LINE 5
UP BLACK LINE 4
BAR LINE
UP BLACK SPACE 4
EIGHTH REST
Q REST
EIGHTH REST
BAR LINE
DN BLACK SPACE 1
FLAG DN 1 LINE 1
DN BLACK SPACE 0
FLAG DN 1 LINE 1
BAR LINE
DN BLACK SPACE 1
DOT
DN BLACK LINE 0
DOT
BAR LINE

BASS CLEF
3 SHARP(S)
DN BLACK SPACE 1
FLAG DN 1 LINE 1
DN BLACK SPACE 0
FLAG DN 1 LINE 1
BAR LINE
DN BLACK SPACE 1
DOT
DN BLACK LINE 0
DOT

Figure 6.7. Partial output of the test samples

Sample III — This is a good example of nicely hand-written music and the recognition rate was better than expected, considering that no autographed music was included in the development samples. Most of the errors were caused by flags too thin to be recognized, for example, the d's in the fifth system, first bar. This error was predictable, since all the flags in the development samples had a minimum thickness that allowed the projection to detect them. One possible way to detect these thin flags would be to take a diagonal projection. The rests were also unrecognized because of the inadequate classifier.

Sample IV — The high quality of print and the standard musical symbols of this sample, printed by the famous Henle Verlag, have resulted in the best recognition performance with a test sample. It should be mentioned that although the symbols used in this sample are similar to many of those found in the development samples, music published by Henle was not included in the development stage. Thus, the result probably reflects the performance level of the program when dealing with music printed by large music publishers.

In general, the use of projections provided an efficient and reliable technique for segmentation, and to a certain extent for classification. Most of the errors arise from the use of a fixed, rectangular height-width plane to classify the symbols. There are two relatively simple steps to improve on the current method. First, rather than using a rectangle, any shape should be allowed to represent a symbol in the plane, depending on the distribution characteristics for each symbol. For example, an ellipsis or a circle can be used. The second step, although more complex, is to allow a symbol to occupy disconnected regions on the plane.

6.4 Conclusions

Although the program is still in its infancy, the respectable recognition rate and speed achieved by simple recognition techniques demonstrate that OMR can indeed be implemented on microcomputer-

based systems. Another strength of the current program, not attained in previous research, is its ability to deal with scores other than those used for its development.

Of course, many improvements and refinements must be made to the program in order to develop practical software. Required enhancements include an increased vocabulary of symbols and some type of learning system. In a typical learning system, the various threshold values are stored in a modifiable database, not hard-coded. When the system makes an error or encounters an unrecognizable symbol, the response provided by a human operator is used to update the database.

A utility should also be devised to allow for manual error-checking of the output. The output can be converted to music notation and compared with the original. This can be done by displaying both versions on the screen or by producing a hardcopy via laser printer. With the latter method, or in a multi-tasking environment, the system may continue to process subsequent pages while the user proofreads. In any case, an audio playback system will expedite the location of any gross errors; this is especially useful when the user is already familiar with the music. A MIDI interface attached to a synthesizer will probably suffice for this purpose, although details that cannot be reproduced through MIDI must still be checked visually.

While preprocessing the input was not required for the samples examined, some form of image restoration must be provided to deal with distorted and noisy images. Finally, the recognition system must be able to process other types of musical settings including orchestral scores and keyboard music. Although all the problems to be encountered in the recognition of polyphonic music cannot be anticipated, it is certain that the projection-based technique can be exploited for this task as well.

Using the experience gained in the present research as a strong starting point, these improvements can be implemented in the near future to create a practical OMR system.

Appendix A – List of Musical Examples

- Bach, J. S. *Six Suites for Solo Cello* (Bryn Mawr: Theodore Presser Company, Pennsylvania, 1964), 50.
- Beckwith, J. *Five Pieces for Flute Duet V* (Toronto: BMI Canada Ltd., 1962), 16.
- Beethoven, L. v. *Sonatas for Pianoforte and Violoncello*. (New York: Schirmer, 1932), 25 (cello part).
- Beethoven, L. v. *Piano Sonata, Op. 111* (Frankfurt: Litolf/Peters, 1978), 326.
- Berg, A. *Wozzeck* (Vienna: Universal Edition, 1955), 479.
- Bozza, E. *Douze Caprices pour Basson* (Paris: Alphonse Leduc, 1968), 9.
- Brahms, J. *Sonata for Piano and Violoncello in F major Op. 99* (Vienna: Wiener Urtext Edition, 1973), 1 (cello part).
- Brod, H. *Études et Sonates pour Hautbois* (Paris: Alphonse Leduc, 1951), 1.
- Cecilia, Sister, Sister John Joseph, and Sister Rose Margaret. *We Sing of Our Land* (Boston: Ginn and Company, 1960), 110.
- Debussy, C. *Syrinx* (Paris: Jobert, 1954), 2.
- Debussy, C. *Préludes, Second Book* (Amsterdam: Brockmans & van Poppel, 1968), 1.
- Goss, J. ed. *Ballads of Britain* (London: John Lane The Bodley Head, 1937), 60.
- Handel, G. F. *The Complete Sonatas for Flute and Basso Continuo* (London: Faber Music Ltd., 1983), 6 (bass part).
- Haydn, J. *Streichquartette. Heft II Op. 9*. (München-Duisburg: Henle Verlag, 1968), 24.
- Hoffmeister, F. A. *Prélude ou Exercice Op. 35 Flûte Solo* (Zurich: Amadeus Verlag, 1979), 10.
- Hotteterre, J. *48 Préludes in 24 Tonarten für Altbloßflöte* (Mainz: Schott, 1972), 21.
- Lasocki, David. (ed.) *Aethology: More Preludes and Voluntaries (England c. 1700) for Treble Recorder Solo* (London: Nova Music, 1981), 3.
- Mozart, W. A. *Thirteen Early String Quartets* (Kassel: Barenreiter, 1966), 7 (violin I part).
- Parrish, I. *Slave Songs of Georgian Islands* (New York: Creative Age Press, 1942), 155.
- Périer, A. *Enseignement Complet de la Clannette* (Paris: Alphonse Leduc, 1931), 2.
- Ravel, M. *Sonate Posthume* (Paris: Editions Salabert, 1975), 17.
- Saygun, A. A. *Partita for Violin Alone, Op. 36* (New York: Southern Music Publishing Co. Inc., 1964), 1.
- Schoenberg, A. *Pelleas und Melisande* (Wien: Universal Edition, 1939), 10.
- Schubert, F. *Das Heimweh, Op. 79/1* (Frankfurt: C.F. Peters, 1956), 149.
- Telemann, G. P. *String Quartet in A major* (Kassel: Hortus Musicus, 1969), 2 (viola part).
- Tippet, M. *Symphony No. 3* (Thetford, Norfolk: Schott & Co. Caligraving Ltd., 1974), 82.
- Tromlitz, J. G. *Sechs Partiten für Flöte* (Frankfurt: Litolf/Peters, 1976), 6.
- Wastall, Peter (ed.) *First Repertoire Pieces for Flute* (London: Boosey & Hawkes, 1982), 23.
- Wolf, H. *Auftrag: Morke Lieder Vol. 5 for High Voice* (New York: Belwin Mills Publishing Corp., n.d.), 63.

Bibliography

- Askenfelt, A. 1976. Automatic notation of played music (status report). *Quarterly Progress and Status Report* Royal Institute of Technology, Speech Transmission Laboratory, Stockholm, Sweden. 1. 1-11.
- Brook, B. S. 1965. The simplified 'Plaine and Easic Code System' for notating music. *Fontes Artis Musicae* 12: 156-60.
- Brook, B. S. ed. 1970. *Musicology and the Computer*. New York: City University of New York Press.
- Buxton, William et al. 1981. Scope in interactive score editors. *Computer Music Journal* 5(3): 50-56.
- Buxton, William et al. 1979. The evolution of the SSSP score editing tools. *Computer Music Journal* 3(4): 14-25.
- Byrd, Donald Alvin. 1984. Music notation by computer. Ph.D. diss., Indiana University.
- Cantor, Don. 1971. A computer program that accepts common musical notation. *Computers and The Humanities* 6(2): 103-9.
- Chafe, C., B. Mont-Reynaud and L. Rush. 1982. Toward an intelligent editor of digital audio recognition of musical constructs. *Computer Music Journal* 6(1): 30-41.
- Erickson, R. 1976. DARMS: A reference manual. Unpublished manuscript.
- Foster, S., W. A. Schloss, A. J. Rockmore. 1982. Toward an intelligent editor of digital audio: Signal processing methods. *Computer Music Journal* 6(1): 42-51.
- Fu, K. S. 1982. *Syntactic pattern recognition and applications*. Englewood Cliffs, NJ: Prentice-Hall.
- Gamble, W. [1923] 1971. *Music engraving and printing*. Reprint. New York: Da Capo Press.
- Handel, G. F. 1746. *Twelve grand concerto for violin & c.* 3d ed. London: Walsh.
- Hermann, G. T. ed. 1979. *Image reconstruction from projections*. Berlin: Springer-Verlag.
- Heussentamm, G. 1987. *The Norton manual of music notation*. New York: W. W. Norton.
- Imai, M. et al. 1984. Automatic transcription of Japanese folk songs. *1984 International Conference on Pattern Recognition* 2: 905-7.
- Kassler, M. 1970. An essay toward specification of a music-reading machine. In *Musicology and the computer*, ed. B. S. Brook, 151-175. New York: The City University of New York Press.
- Kassler, M. 1972. Optical character recognition of printed music: A review of two dissertations. *Perspective of New Music* 11: 250-59.
- Knowlton, Prentiss H. 1971. Interactive communication and display of keyboard music. Ph.D. diss., University of Utah, Salt Lake City.
- Kolb, Randall Martin. 1984. A real-time microcomputer-assisted system for translating aural, monophonic tones into music notation as an aid in sight-singing. Ph.D. diss., Louisiana State University and Agricultural and Mechanical College.
- Krummel, D. W. 1975. *English music printing 1553-1700*. London: The Bibliographical Society.
- Lee, Myung Woo and Jong Soo Choi. 1985. The recognition of printed music score and performance using computer vision system. (in Korean) *Journal of Korea Institute of Electronics Engineers* 22(Sept.): 10-16.
- Lockwood, L. 1970. A stylistic investigation of the masses of Josquin Desprez with the aid of the computer: A progress report. In *Musicology and the computer*, ed. B. S. Brook, 19-27. New York: The City University of New York Press.

- Maxwell III, J. T. and S. M. Ornstein. 1984. Mockingbird: A composer's amanuensis. *Byte* 9(1): 383-401.
- Mercuri, R. T. 1981a. MANUSCRIPT: Music notation for the Apple II. *Proceedings of the Symposium on Small Computers in the Arts* 8-10.
- Mercuri, R. T. 1981. Music editors for small computers: A comparative study. *Creative Computing* 7(Feb.): 18-19.
- Miller, Jim. 1985. Personal composer. *Computer Music Journal* 9(4): 27-37.
- Moore, J. A. 1975. On the segmentation and analysis of sound by digital computer. Ph.D. Thesis, Stanford University. 1975.
- Moore, J. A. 1977. On the transcription of musical sound by computer. *Computer Music Journal* 1(4): 32-38.
- Nagel, R. N. and A. Rosenfeld. 1972. Ordered search technique in template matching. *Proceedings to IEEE* 60: 242-44.
- Nakano, Y. et al. 1973. Improvement of Chinese character recognition using projection profiles. *1973 Proceedings of the International Joint Conference on Pattern Recognition*: 172-78.
- Ohteru, S. and T. Matsushima. 1985. Automatic recognition of printed music. *Journal of the Acoustical Society of Japan* 41(June): 412-15.
- Piszcalski, M. and B. Geller. 1979. Computer analysis and transcription of performed music: A project report. *Computers and Humanities* 13: 195-206.
- Piszcalski, M. and B. Geller. 1977. Automatic music transcription. *Computer Music Journal* 1(4): 24-31.
- Piszcalski, M. et al. 1981. Performed music: Analysis, synthesis, and display by computer. *Journal of Audio Engineering Society* 29: 38-46.
- Piszcalski, M. and B. Geller. 1982. A computational model of music listening. *1982 Proceedings of Graphics Interface* 89-96.
- Piszcalski, M. 1986. A computational model of music transcription. Ph.D. diss., University of Michigan.
- Poole, H. E. 1980. Printing and publication of music. *The new Grove dictionary of music and musicians*, edited by S. Sadie, 20 vols. London: Macmillan, 15: 232-274.
- Prerau, D. S. 1970. Computer pattern recognition of standard engraved music notation. Ph.D. diss., MIT.
- Pruslin, D. H. 1966. Automatic recognition of sheet music. Sc.D. diss., MIT.
- Raskin, J. 1980a. Using the computer as a musician's amanuensis. part one: Fundamental problems. *Byte* 5(April): 18-28.
- Raskin, J. 1980b. Part two: Going from keyboard to printed score," *Byte* 5(May): 120-8.
- Read, G. 1969. *Music Notation*. 2d ed. Boston: Allyn and Bacon.
- Roads, C. 1986. The Tsukuba musical robot. *Computer Music Journal* 10(2): 39-43.
- Ross, T. 1970. *The Art of Music Engraving and Processing*. Miami: Hansen Books.
- Schmid, C. E. 1977. Acoustic pattern recognition of musical instruments. Ph.D. diss., University of Washington.
- Schnell, C. 1986. The input of musical information to the computer. (in German) *Output* 15(May): 51-60.
- Shyu, R. 1986. A new score input method for computer music. *Journal of the Acoustical Society of America* 80(S1): S87.
- Smith, L. 1973. Editing and printing music by computer. *Journal of Music Theory* 17: 292-309.

- Talbot, A. 1983. Finished musical scores from the keyboard. *Proceedings of the ACM Annual Conference*: 234–37.
- Tojo, Akio, et al. 1982. Automatic recognition system for music scores. *Proceedings of International Conference on Pattern Recognition 2*: 1223.
- Tucker, W. H. et al. 1977. An interactive aid for musicians. *International Journal of Man-Machine Studies* 9: 357–58.
- Yavelow, C. 1985. Music software for the Apple Macintosh. *Computer Music Journal* 9(4): 52–67.