Machine Vision and Applications (1993) 6:140–150

Machine Vision and Applications

MUSER: A prototype musical score recognition system using mathematical morphology

Bharath R. Modayur, Visvanathan Ramesh, Robert M. Haralick, and Linda G. Shapiro

Intelligent Systems Laboratory, Electrical Engineering Department, FT-10, University of Washington, Seattle WA 98195, USA

Abstract. Music representation utilizes a fairly rich repertoire of symbols. These symbols appear on a score sheet with relatively little shape distortion, differing from the prototype symbol shapes mainly by a positional translation and scale change. The prototype system we describe in this article is aimed at recognizing printed music notation from digitized music score images. The recognition system is composed of two parts: a low-level vision module that uses morphological algorithms for symbol detection and a high-level module that utilizes prior knowledge of music notation to reason about spatial positions and spatial sequences of these symbols. The high-level module also employs verification procedures to check the veracity of the output of the morphological symbol recognizer. The system produces an ASCII representation of music scores that can be input to a music-editing system. Mathematical morphology provides us the theory and the tools to analyze shapes. This characteristic of mathematical morphology lends itself well to analyzing and subsequently recognizing music scores that are rich in well-defined musical symbols. Since morphological operations can be efficiently implemented in machine vision systems that have special hardware support, the recognition task can be performed in near real-time. The system achieves accuracy in excess of 95% on the sample scores processed so far with a peak accuracy of 99.7% for the quarter and eighth notes, demonstrating the efficacy of morphological techniques for shape extraction.

Key words: Mathematical morphology – Low-level vision – Feature extraction – Connected components – Music notation

1 Introduction

The performance of efficient algorithms to recognize isolated two-dimensional shapes deteriorates rapidly as the shapes become increasingly interconnected. Music score representation is one such domain where the different symbols are arranged in a truly two-dimensional fashion. Utilization of prior knowledge about music representation becomes necessary to devise efficient methods to recognize the music symbols.

A musical-score layout is the visual manifestation of interrelated properties of musical sound – pitch, intensity, time, timbre and pace (Read 1969). Symbols indicating the tones, their duration and the manner of performance form the music notation. The basic symbols we are concerned with in this paper include the:

- Staff – an arrangement of five parallel lines together with spaces between them
- Clefs – symbols that determine the pitch associated with a particular staff
- Note heads and stems – symbols that determine a note's time value
- Flags and beams – other symbols that determine the time value of a note
- Rests and pauses – symbols that indicate moments of silence in a musical piece
- Accidentals and key signatures – symbols that are placed in front of a note to modify its pitch, i.e., to raise or lower the pitch
- Barlines – thin vertical lines drawn through the staff to set off the time length of each measure

Mathematical morphology provides us the theory and the tools to analyze shapes. Printed music score is rich in well-defined, but interconnected shapes. The shape-based approach of mathematical morphology thus lends itself well to being used as an effective tool in understanding printed music score.

The prototype system proceeds through three distinct stages to recognize music scores. They are: **layout extraction, symbol recognition**, and **high-level reasoning**. In the first stage, the skew and the scale of the image is determined by locating the position of the staff lines. This is achieved by a combination of morphological and image processing techniques. In the second stage the different symbols are extracted using sequences of morphological operations. Prior knowledge of music representation is utilized in making the detection algorithms simple as well as efficient. In the last stage, prior knowledge of music representation (Read 1969)

Correspondence to: Robert Haralick

and the shapes of the symbols is made use of to reason about the spatial positions and sequences of those symbols. This high-level module also employs verification procedures to check the veracity of the output from the morphological symbol recognizer (Fig. 1). The remainder of the paper is organized as follows. Section 2 outlines the problem we are trying to solve and the final goal. Section 3 briefly discusses related work. Section 4 gives a basic introduction to the morphological operations and the notations used. Section 5 outlines the prior assumptions. Sections 6 and 7 discuss the layout extraction and the symbol recognition strategy. Section 8 gives a detailed look into a few of the representative algorithms for the symbol detection stage. Section 9 describes the high-level reasoning module and explains the postprocessing for a few important classes of symbols. Section 10 discusses the experiments and the results obtained so far.

2 The problem and the goal

We have a known representation of music, an image of printed score sheet. We also have prior knowledge of the various symbols that can appear on the score, the set of possible positions different symbols can occupy, and some definite sequences of symbols. The problem is then to recognize all the meaningful symbols on the image, their spatial positions, and the spatial sequence of these symbols (see Modayur 1991 for an earlier version of this work). If there are N symbols $S_1, S_2, ...S_N$ which occur in the image in unknown locations, we can represent the image as $I = \bigcup_{i=1}^{N} (S_i)_{x_i,y_i}$, where the S_i do not intersect (this condition is relaxed for symbols like staff line and note stems which intersect other symbols). The purpose of the morphological sequence of operations is to extract the individual symbols with their respective translations. Let us assume that the first symbol, S_1 has been extracted and removed from I. Now, the residue image I_{res} can be represented as $I_{res} = \bigcup_{i=2}^{N} (S_i)_{x_i,y_i}$. The symbol extraction proceeds until I has been decomposed completely.

The goal is to represent our recognition result as a sequence of text symbols as follows:

3C4 3D8 [4C4 4E4]

This representation indicates a C note in the third octave (this is the C below middle C) that is a quarter note, followed by a D note in the third octave that is an eighth note. Then there is a chord consisting of a quarter note C in the fourth octave and a quarter note E in the fourth octave.

Since MUSER is designed to work on printed forms of music representation, it enjoys the advantage of encountering symbols of regular shape. In this regard the problem is well defined. Once the image is normalized (which can be done if the spacing between the staff lines is known) the relative sizes of the various symbols can be determined. This is helpful especially in creating the appropriately sized structuring elements.

3 Related work

The doctoral dissertations of Pruslin (1966) and Prerau (1970) used contour-tracing techniques for symbol segmentation and hence staff line separation was problematic. The gaps created as a result of staff line removal were filled and the aspect ratio of segmented symbols along with some music rules were used to recognize the symbols. Fujinaga's thesis work (1988) used projection profile techniques for staff line detection as well as symbol segmentation and recognition. The projection techniques tolerate skew in the staff line orientation. This work assumed that the sizes of the different music symbols in a given score sheet are relative. The results were reported for four carefully chosen score samples. The recognition accuracy was 73% on the average.

The ENGRAVE system (for handwritten music score recognition) reported in Roach and Tatem (1988) exploited the fact that music representation is a highly structured domain and utilized knowledge of music to achieve significant improvements in low-level image processing to extract primitive features. The system processed six images of handwritten scores and the results were reported to be good. The system reported by Baumann and Dengel (1987) used a top-down strategy in order to take advantage of the hierarchical structure of music representation. Using music rules and constraints, it applied a decision tree classifier to recognize music symbols with an accuracy of about 90%.

MUSER utilizes music rules and constraints to some extent in the low-level symbol extraction stage and to a large extent in the high-level reasoning stage. The low-level stage uses morphological sequences in a *feature extraction–topology matching* strategy (to be explained later) to detect the musical symbols. Staff line extraction, which is a lengthy and often troublesome process in other systems, is achieved by a single morphological operation. The high-level module uses prior knowledge about music representation and, using the output of the symbol recognizer, generates an ASCII representation of the music score. This module also employs verification procedures to check the veracity of the low-level output. The overall symbol recognition accuracy of the system is in excess of 95%, with an accuracy of 99.7% for the quarter and eighth notes.

4 Morphological operations and notations

There are five basic morphological operations that are used throughout this work. In this section, we will give a brief description of the operations and their mathematical notation (see Haralick et al. 1987 for a more complete description).

4.1 Dilation

Dilation is the morphological transformation which combines two sets using vector addition of set elements.

Let A and B be subsets of E^N. The dilation of A by B is denoted by $A \oplus B$ and is defined by

$$A \oplus B = \{c \in E^N \mid c = a + b \text{ for some } a \in A \text{ and } b \in B\}.$$

In practice, the set A is considered as the image undergoing transformation and the set B is referred to as the structuring element. To dilate an image, we run the structuring element origin over all the binary one pixels of the image. The area swept by the structuring element is the dilated image.

4.2 Erosion

Erosion, the morphological dual of dilation, is a shrinking operation. It is the morphological transformation which combines two sets using vector subtraction of set elements.

Let A and B be subsets of E^N. The erosion of A by B is denoted by A $\ominus$ B and is defined by

$$A \ominus B = \{x \in E^N \mid x + b \in A \text{ for every } b \in B\}.$$

To erode an image, we run the structuring element origin over all the pixels in the image. We mark those pixels at which the structuring element origin can stand and where its entire area covers only binary one pixels. The area of marked pixels is the eroded image.

4.3 Opening

The opening of an image A by a structuring element K is denoted by $A \circ K$ and is defined as $A \circ K = (A \ominus K) \oplus K$.

The opening operator runs the structuring element through the area of binary one pixels, keeping the area of the structuring element contained in the area of binary one pixels in the image. The area swept by the structuring element is the opened image. Opening an image with a disk structuring element smooths the boundary, breaks narrow isthmuses, and eliminates small specks.

4.4 Closing

The closing of an image A by a structuring element K is denoted by $A \bullet K$ and is defined as $A \bullet K = (A \oplus K) \ominus K$. Closing is the morphological dual of opening. What opening does to the foreground, closing does to the background. Closing an image with a disk-structuring element smooths the boundary, fuses narrow breaks, fills small holes, and fills gaps on the boundary.

4.5 Hit-or-miss transformation

The hit-or-miss transformation of an image A by two structuring elements K_1 and K_2, $K_1 \cap K_2 = \emptyset$, can be defined as

$$A \otimes (K_1, K_2) = (A \ominus K_1) \cap (A^C \ominus K_2) .$$

The hit-or-miss transformation is useful in locating patterns in an image. The structuring element K_1 is used to define the foreground pattern being sought, while the structuring element K_2 is used to define the neighboring background pattern (refer to Costa 1990 for a detailed description of various morphological sequences for pattern detection).

4.6 Structuring elements

The morphological symbol recognizer uses, among other shapes, three structuring elements of standard shape: (1) line-, (2) disk-, and (3) box-structuring elements. Their notations are as follows.

- A line-structuring element with origin at (0, 0) and extending from (x_1, y_1) to (x_2, y_2) is denoted by $line(x_1, y_1, x_2, y_2)$
- A disk-structuring element with its origin at the center and radius r is denoted by $disk\ r$
- A box-structuring element with width w and height h is denoted by $box(w, h)$

4.7 Image assignment notation

The equation

$$I_{original} = I_{original} \oplus disk\ 5$$

would mean the following. The original image $I_{original}$ is to be dilated by a disk of radius 5 pixels and the resulting image is to replace the original image.

5 Prior assumptions

The following assumptions are made about the structure of music scores that MUSER can handle.

- The sizes of the different symbols are relative. This essentially means that once the scale of the image is computed, an approximate estimate of the sizes and shapes of the various symbols can be determined.
- The staff lines are equally spaced. The layout extraction module does a median calculation on the spacing between the horizontal staff lines to determine the scale of the image. Thus, we require the staff lines to be equally spaced.
- The image does not have a large skew. This is required to make the staff extraction algorithm remain simple. The staff is extracted by opening the image with a long horizontal line (about 40 pixels wide). This would detect horizontal and near horizontal lines (if the lines are more than 1 pixel thick). If there is a large skew, the algorithm will fail to detect the staff lines.

The following assumptions are made about the representation of music scores that MUSER is designed to handle.

1. The number of parallel lines in a staff is assumed to be five. This means that we do not allow for percussion staves, cue staves, etc.
2. We will deal with multiple voice musical scores.
3. Current implementation assumes that the printed musical score follows some of the conventions specified by Roemer (1985).

4. The distance between successive note heads in pixels is proportional to the number of pixels in the measure and the left note duration.
5. Accidentals are placed squarely on the line or the space directly in front of the note they alter. Our implementation just looks for notes in the bottom right quadrant of the accidental while associating accidentals with notes.
6. Stems, in general, go down when attached to the left of the note. They go up when attached to the right of the note. The stem length is normally the length of one octave.

There are a number of other conventions about rest symbols and augmentation dots that our system uses. For example, a quarter rest is assumed to be centered on the staff. A half rest touches the third staff line above, while a whole rest touches the fourth staff line below. Some of the assumptions mentioned above provide answers for the choices of thresholds used in the postprocessing stage that filters out spurious symbols that are detected in the feature extraction step.

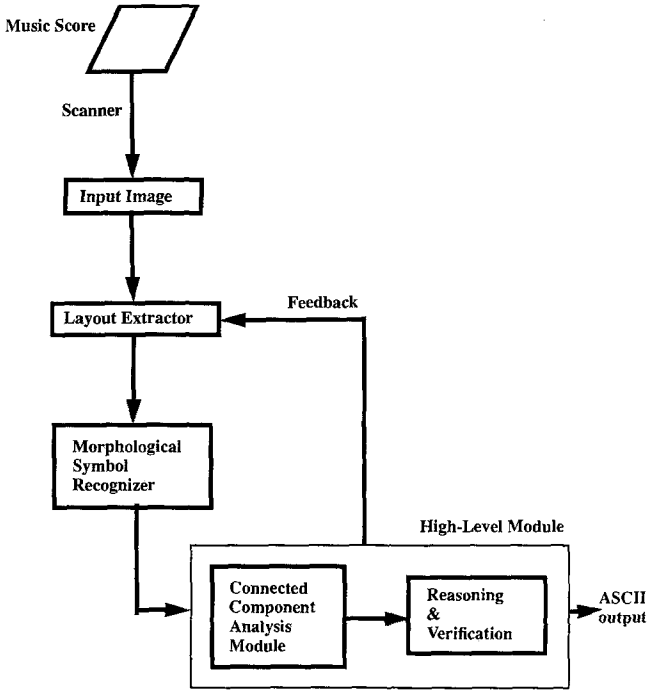


Fig. 1. MUSER – system block diagram

6 Layout extractor

The first step that MUSER goes through in the recognition process is determining the scale and the skew of the image. The skew in the image causes the rectangular coordinate system, relative to which the different symbols are located, to be rotated. This rotation can be corrected once the skew has been determined. The scale of the image is used to fix the sizes of the different standard size structuring elements used by the symbol recognizer. The scaling of the standard structuring elements can be achieved in a semi-interactive way, provided polygonal approximations of the original structuring elements are available.

To extract the staff lines, the image is opened with a horizontal line 35 pixels wide. Let this line structuring element be **K**.

$$I_{staff} = I_{input} \circ \mathbf{K}$$

A connected component analysis on I_{staff} would give information about the extremities of the horizontal lines in the image and their orientation. The median of the spacing between adjacent lines can now be computed. This would give an approximate estimate of the scale of the image. Let this estimated scale be denoted by S_I. The dimensions of all the structuring elements that are used for symbol extraction would then be scaled by this value. The orientation of the lines would indicate the skew of the image.

7 Morphological symbol recognizer

A preliminary examination of the symbols that occur in music representation reveals the following. There are some symbols whose features are exactly similar to (or in some cases, a subset of) others. The only factor that distinguishes the former from the later may be the size. A typical example of this fact is the grace note. The only thing that distinguishes a grace note from a regular note is its size. This overlap of features among the different symbols necessitates the use of a prudent sequence of symbol extraction.

How do we decide which symbols to extract earlier and which ones later? The guideline that is employed in MUSER in deciding the priority of a symbol in the extraction sequence is its foreground area and the lack of distinguishing concavities and curvatures. Symbols like the repeat measure bar, repeat symbol, clef, and full note heads would thus get a higher priority while the rest of the symbols, accidentals, and others that have distinguishing concavities, would come later in the symbol extraction sequence. A regular note would thus have to be extracted before the grace notes since we would be using similarly shaped structuring elements (but of different sizes) to extract both the symbols.

The algorithms used for symbol detection follow a two-step process. In the first step, the primitive features of a given symbol are extracted. The second step utilizes prior knowledge of the topological relationship among the different features to detect the symbol. Please note that the structuring elements employed throughout this symbol detection phase would "loosely" follow the shape of the medial axis (the skeleton) of the feature shape being sought. This is to incorporate a certain degree of tolerance in the detection process. Thus, imperfections such as a few missing foreground pixels, broken edges, and blurred corners will not affect the output of the symbol detection process. Algorithms that utilize the background features of symbols are less sensitive (less tuned) to structuring element sizes than the ones that use foreground features. Symbols like the notehead, however, do not possess background features and consequently

are dependent on suitably sized structuring elements for their extraction. This is where the relative-size assumption comes in handy. We assume that with the scale of the image determined (by the staff-line distance), the structuring element sizes can be fixed. This still does not imply that the structuring element sizes have no room for tolerance. The staff line extraction works as adequately with a horizontal line 20 pixels wide as with the 35-pixel line that we used in our experiments. The notehead extraction still functions well with a disk-structuring element (7-pixel radius) as with the elliptical structuring element we used.

Once a symbol is detected, if it is local in nature it is removed from the input image. This is to prevent the extracted symbol from giving positive results in subsequent processing. Exceptions to this are the staff lines and note stems which are not removed after detection.

In the following section, we will discuss a few of the representative algorithms used in the feature extraction–topology matching strategy to detect different symbols. For a more complete discussion on the algorithms see Modayur and Haralick (1991).

8 The symbol detection algorithms

8.1 Clef symbols

The only clef symbols that MUSER handles currently are the bass and treble clefs.

8.1.1 Bass clef

The bass clef is the last symbol in our symbol detection sequence. For convenience, we discuss it here. The important feature here is the head of the bass clef symbol with two dots (one below the other) to the right (Fig. 2). The head portion is easily detected by opening the input image with a disk-shaped structuring element K_{bass} (of $0.44S_I$ pixel radius) with a downward tail (see Fig. 8).

$$I_{bassHead} = I_{input} \circ K_{bass}$$

Since K_{bass} has gaps in it, $I_{bassHead}$ is closed with a box to fill the gaps. This produces a single connected component.

$$I_{bassHead} = I_{bassHead} \bullet box(1.67\mathbf{S_I}, 0.33S_I)$$

The result of this operation is illustrated in Fig. 3. Now, the two dots lying to the right of the head have to be extracted. To begin with, all the horizontal and vertical lines in the image are removed.

$$I_h = I_{input} \circ line(0, 0, 1.11\mathbf{S_I}, 0)$$
$$I_v = I_{input} \circ line(0, 0, 0, 1.11\mathbf{S_I})$$

Remove I_h and I_v from the image.

$$I_{temp} = I_{input} \cap (I_h \cup I_v)$$
$$I_{noLines} = I_{input}\ XOR\ I_{temp}$$

Figure 4 shows the image after the above operation.

To locate the individual dots, the resultant image is opened with a disk. This should be followed by another opening to get rid of blobs bigger than the required size.

$$I_{dots} = I_{noLines} \circ disk0.11\mathbf{S_I}$$

Then we get rid of the bigger blobs.

$$I_{bigBlobs} = I_{dots} \circ line(0, 0, 0.44\mathbf{S_I}, 0)$$
$$I_{temp} = I_{bigBlobs} \cap I_{dots}$$
$$I_{dots} = I_{dots}\ XOR\ I_{temp}$$

Since we seek a pair of dots with one vertically below the other separated by about $\mathbf{S_I}$ pixels, the image I_{dots} is closed with a vertical structuring element.

$$I_{dotPair} = I_{dots} \bullet line(0, 0, 0, 1.11\mathbf{S_I})$$
$$I_{dotPair} = I_{dotPair}\ XOR\ I_{dots}$$

The second operation produces results only at those places where pairs of dots occur, one vertically below the other and separated by less than $1.11\mathbf{S_I}$ pixels. This is illustrated in Fig. 5. The required features have now been extracted. What follows is another sequence of morphological operations to check the topology. Since the dot pair is expected to lie to the right of the bass head portion, the head is translated to the right and intersected with the dot pair.

$$I_{dilateRight} = I_{bassHead} \oplus line(0.83\mathbf{S_I}, 0, 1.11\mathbf{S_I}, 0)$$
$$I_{bass} = I_{dilateRight} \cap I_{dotPair}$$

The result of the above operation is shown in Fig. 6. The operation produces results in places where bass clefs are not present. In the actual symbol extraction sequence, bass clefs are extracted last. Thus, the other symbols would have been removed by then and the bass clef extraction results in foreground pixels only at those locations where the bass clef symbols are present. The following sequence is employed to get rid of the bass symbol from the input image.

$$I_{bassDil} = I_{bass} \oplus disk0.22\mathbf{S_I}$$
$$I_{dilateLeft} = I_{bassDil} \oplus line(0, 0, -1.67\mathbf{S_I}, 0)$$
$$I_{dilateDown} = I_{dilateLeft} \oplus line(0, 0, 0, -1.67\mathbf{S_I})$$
$$I_{temp} = I_{dilateDown} \cap I_{input}$$
$$I_{input} = I_{temp}\ XOR\ I_{input}$$

Since I_{bass} would produce output at the location of the dot pair to the right of the bass symbol, I_{bass} is dilated to the left and then down. The intention is to cover the entire bass symbol to facilitate later removal. The result of the three dilations is then removed from the input image in the usual way.

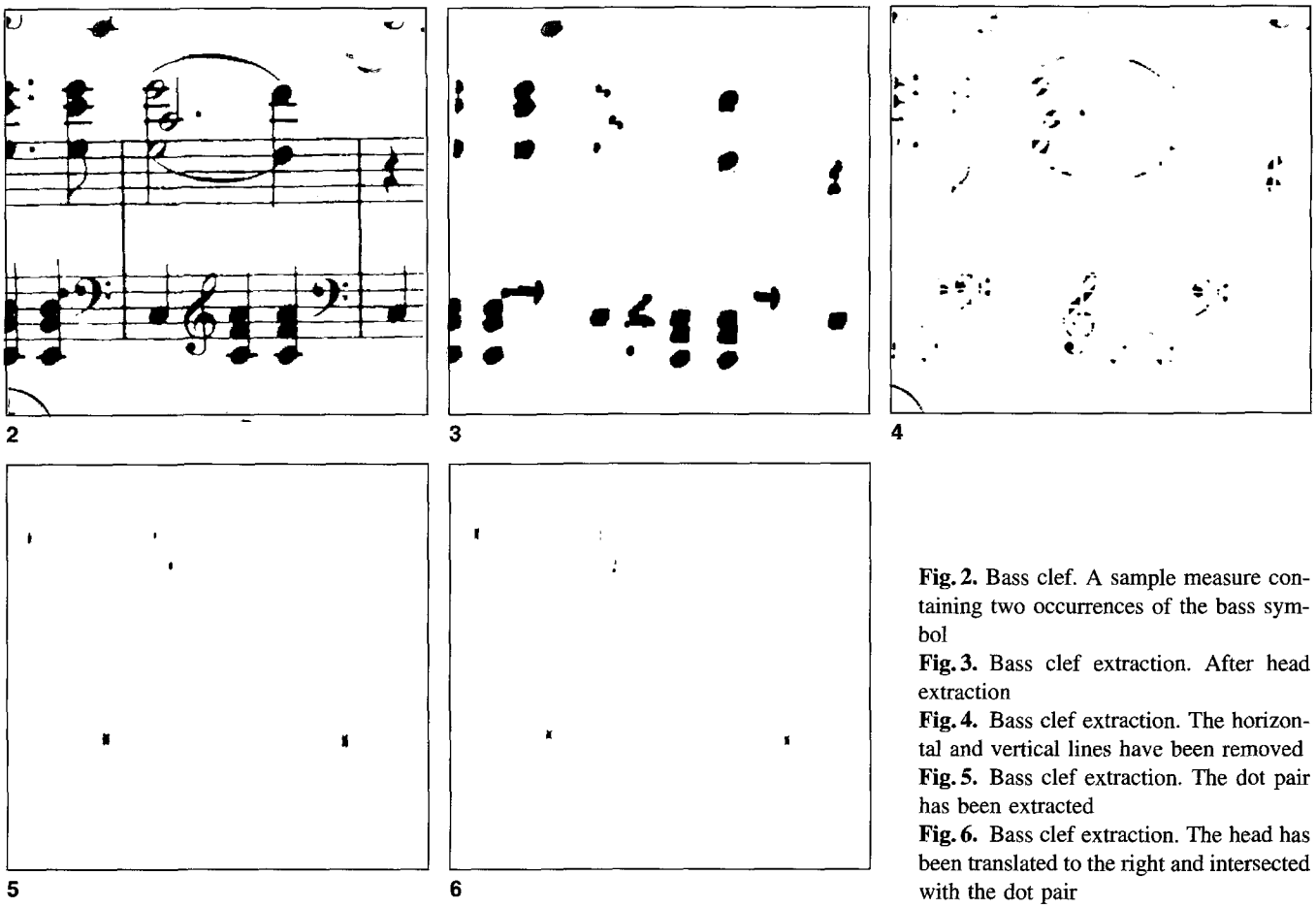


Fig. 2. Bass clef. A sample measure containing two occurrences of the bass symbol

Fig. 3. Bass clef extraction. After head extraction

Fig. 4. Bass clef extraction. The horizontal and vertical lines have been removed

Fig. 5. Bass clef extraction. The dot pair has been extracted

Fig. 6. Bass clef extraction. The head has been translated to the right and intersected with the dot pair

8.1.2 Treble clef

The treble clef symbol (Fig. 7) is detected by a sequence of openings followed by an intersection. To begin with, the near-circular shape forming a part of the body of the treble symbol is extracted. This is achieved by opening the input image with a semi- circular structuring element $K_{treble1}$ as shown in Fig. 8.

$$I_{treble1} = I_{input} \circ K_{treble1}$$

The body of the treble clef symbol is extracted by opening with a box-shaped structuring element $K_{treble2}$ that is $0.44\mathbf{S_I}$ pixels wide and $1.8\mathbf{S_I}$ pixels tall, oriented roughly at a 45° angle.

$$I_{treble2} = I_{input} \circ K_{treble2}$$

The output of the last opening operation would yield the body of the treble symbol that lies approximately at the middle, with the other feature lying below it. The following sequence is employed to verify the spatial relationship of the two features extracted so far. The structuring elements used are K_{line1} = line $(0,0.55\mathbf{S_I},0,1.11\mathbf{S_I})$ and K_{line2} = line $(0,0,-1.11\mathbf{S_I},0)$.

$$I_{dilateUp} = I_{treble1} \oplus K_{line1}$$

The structuring element K_{line1} does not include the origin. Thus the above operation translates $I_{treble1}$ upwards by $0.55\mathbf{S_I}$ pixels and then dilates the translated image by $0.55\mathbf{S_I}$ pixels vertically.

$$I_{dilateLeft} = I_{dilateUp} \oplus K_{line2}$$

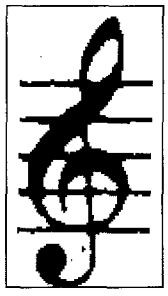
This operation dilates the result of the previous dilation to the left. This image is now intersected with $I_{treble2}$, which would contain the body of the treble clef symbol.

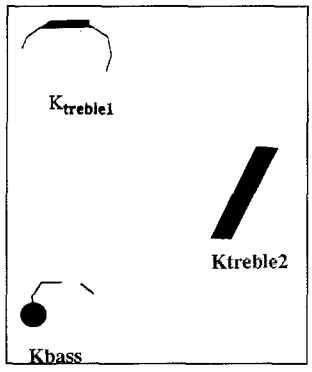
$$I_{treble} = I_{dilateLeft} \cap I_{treble2}$$

The image I_{treble} would then have foreground pixels only at the locations of the treble clef symbols that occurred in the input image. The treble clef symbols could potentially appear in two different sizes, the smaller of which is used to denote clef changes inside a measure. This smaller treble clef symbol is approximately 80% of the regular size. Thus, the small treble clef is extracted by an exactly similar procedure but with all the structuring elements scaled down by a factor of 0.8.

Once the treble clef symbol is detected, it is removed from the input image by the following sequence:

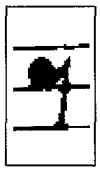
$$I_{temp} = I_{input} \cap I_{treble}$$
$$I_{input} = I_{input}\ XOR\ I_{temp}$$

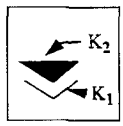

7



8


9

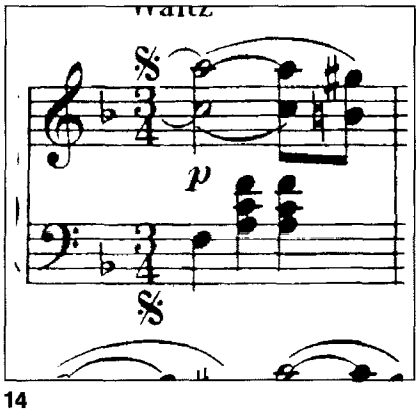

10



11


12


13


14

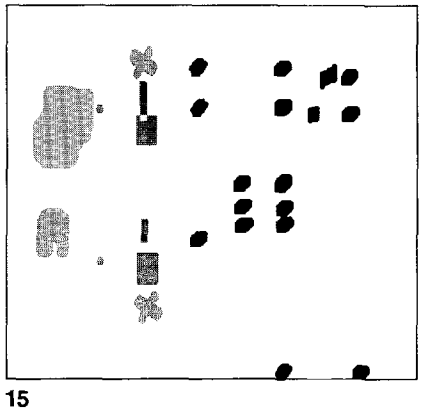

15

Fig. 7. Treble clef
Fig. 8. Structuring elements for clef symbols
Fig. 9. Filled note head
Fig. 10. Eighth rest
Fig. 11. Hit- or miss structuring elements for eighth rests
Fig. 12. Note beams
Fig. 13. Quarter rest
Fig. 14. original image of a sample measure
Fig. 15. Recognized symbols of the above measure with the symbols overlaid

8.2 Filled note heads

Filled note heads (Fig. 9) are present in quarter, eighth, and sixteenth notes. The structuring element K_{blob} that is used to extract these note heads is elliptical, with a major axis to minor axis ratio of approximately 5:4. The ellipse is oriented roughly at a 45° angle.

$$I_{blob} = I_{input} \circ K_{blob}$$

This opening would extract the noteheads of quarter, eighth and sixteenth notes (and other lower duration notes, if present). As before, the extracted note heads are removed from the input image to prevent interference of this feature with subsequent feature extraction sequences.

$$I_{temp} = I_{input} \cap I_{blob}$$
$$I_{input} = I_{temp}\ XOR\ I_{input}$$

8.3 Eighth rest

A careful inspection of the eighth rest symbol (Fig. 10) reveals that there are at least two features one can look for. This rest symbol has a disk-shaped head with an up-concavity to its right. The head is detected simply by opening with a disk-shaped structuring element. The up-concavity is detected by a hit-or-miss transform.

$$I_{head} = I_{input} \circ disk0.17\mathbf{S_I}$$

The above sequence extracts the head of this symbol. Next, the up-concavity is located by hit-or-miss transform. The structuring elements used are: K_1, which is wedge shaped with a thin body, and K_2, which is also a wedge but with a thicker body, as shown in Fig. 11.

$$I_{upConcavity} = I_{input} \otimes (K_1, K_2)$$

Now, the following sequence is used to check whether the head and the up-concavity lie in the expected relative spatial position.

$$I_{leftDil} = I_{upConcavity} \oplus line(0, 0, -0.28\mathbf{S_I}, -0.5\mathbf{S_I})$$

This operation dilates the concavity output in a left-downward direction. Intersection of this dilated concavity with the head would yield the eighth rest.

$$I_{rest8} = I_{leftDil} \cap I_{head}$$

The extracted eighth rest is then removed from the input image.

$$I_{temp} = I_{input} \cap I_{rest8}$$
$$I_{input} = I_{temp}\ XOR\ I_{input}$$

8.4 Note stems

The half notes, quarter notes, eighth notes and sixteenth notes all have vertical stems supporting the corresponding note heads. Extraction of these note stems is achieved by opening with a vertical structuring element about $2.22\mathbf{S_I}$ pixels long.

$$I_{stem} = I_{input} \circ line(0, 0, 0, 2.22\mathbf{S_I})$$

This opening obviously would extract all vertical lines that are $2.22\mathbf{S_I}$ or more pixels long. Thus, in addition to the note stems we would also be extracting portions of other symbols. Intersection of these stems with the note heads, however, would get rid of all the spurious note stems. This is achieved by the high-level module when it reasons about topology.

8.5 Note beams

Thick beam lines (Fig. 12) connecting two or more note stems identify an eighth or sixteenth note or other lower duration notes. For an eighth note there is a single beam line, for a sixteenth note two beam lines, and so on.

To locate a beam line that joins two or more notes or that which protrudes from a single stem, the following fact is utilized: The beam lines could be oriented in just a fixed number of orientations. In the sample music score we worked with, there were just two such orientations. If the set of all possible orientations the beam lines could assume is known, they could be extracted by opening with thick straight lines (boxes) at those orientations and taking the union of all those openings. To locate the thick beam lines joining two or more note stems, opening is employed with boxes that were oriented roughly at $+11°$ and $-11°$. The structuring element K_{boxUp} is about $2\mathbf{S_I}$ pixels wide, $0.2\mathbf{S_I}$ pixels tall, and oriented approximately at $+11°$. The other structuring element $K_{boxDown}$ is of the same dimensions but oriented approximately at $-11°$.

$$I_{slantUp} = I_{input} \circ K_{boxUp}$$
$$I_{slantDown} = I_{input} \circ K_{boxDown}$$

Now, these note beams are removed from the input image.

$$I_{temp} = I_{input} \cap (I_{slantUp} \cup I_{slantDown})$$
$$I_{input} = I_{temp}\ XOR\ I_{input}$$

8.5.1 Small note beams

After the note beams that join note stems have been detected, the relatively smaller note beams that protrude from single note stems are targeted. These smaller note beams could either lie above or below the larger note beams. The following sequence is used to locate the small note beams. A box-structuring element $K_{smallSlant}$ that is $0.88\mathbf{S_I}$ pixels wide, $0.39\mathbf{S_I}$ pixels tall, and oriented approximately at $+11°$ is used.

$$I_{smallSlant} = I_{input} \circ K_{smallSlant}$$
$$I_{bigSlant} = I_{slantUp} \cup I_{slantDown}$$
$$I_{bigSlant} = I_{bigSlant} \oplus line(0, 0, 0, 1.67\mathbf{S_I})$$
$$I_{bigSlant} = I_{bigSlant} \oplus line(0, 0, 0, -1.67\mathbf{S_I})$$
$$I_{smallSlant} = I_{smallSlant} \cap I_{bigSlant}$$

The first operation opens the input image with the small box-structuring element $K_{smallSlant}$. The small note beam, as we know, lies either above or below the larger note beam. The result of the following sequence of three operations is the larger note beams dilated downward and upward. This dilated image defines regions above and below the larger note beams, within which the smaller note beams could be expected. The last operation in the sequence intersects the result of the first opening operation with this dilated image. This sequence thus extracts the small note beams lying either above or below the large note beams.

It is possible that the above sequence yields spurious beam lines (though they have not in our experiments so far) that do not protrude from note stems. Hence, only those note beams that emerge from note stems should be extracted. The same reasoning is not employed in extracting the larger note beams (in the previous section) because we expect larger features to be more reliable than the smaller ones.

$$I_{trueSlants} = I_{smallSlant} \cap I_{stem}$$
$$I_1 = I_{trueSlants} \oplus disk0.22\mathbf{S_I}$$
$$I_1 = I_1 \oplus line(0, 0, 0.55\mathbf{S_I}, 0)$$
$$I_{smallSlant} = I_1 \oplus line(0, 0, -0.55\mathbf{S_I}, 0)$$

The first operation in the above sequence intersects the small beam lines with the note stems to remove spurious beam lines. Since this operation yields only the endpoints of the small beam lines (in the vicinity of the note stems), the result of the first operation is dilated with a disk and then allowed to expand horizontally. The following three dilations achieve this. The extracted symbol is removed from the input image in the usual way.

$$I_{temp} = I_{input} \cap I_{smallSlant}$$
$$I_{input} = I_{temp}\ XOR\ I_{input}$$

8.6 Quarter rest

Symbols that denote different durations of rest always occupy the same vertical position relative to the staff lines. The distinguishing features of a quarter rest are its concavities (Fig. 13). It has a left concavity, a right concavity, and a down concavity. The geographical positions of the concavities with respect to each other is also an important distinguishing factor. The right concavity is to the right of the left concavity and the down concavity is below both the left and the right concavities. To locate a particular concavity, say the right concavity, the image is opened with a wedge-shaped structuring element $K_{rightWedge}$ with its acute or inner side facing right (like a $<$ symbol). A hit-or-miss transformation on the resulting image locates the right concavity. The structuring elements in this case would be K_1, which would be exactly the same as $K_{rightWedge}$ to detect the foreground pattern, and K_2, which would be a filled triangle suitably

positioned with respect to K_1 to detect the background concavity.

$$I_{open} = I_{input} \circ K_{rightWedge}$$
$$I_{rightConc} = I_{open} \otimes (K_1, K_2)$$

A similar approach is used to detect the left concavity. Relative positions of the concavities can then be used to detect the presence of the quarter rest.

9 High-level reasoning module

The output of the morphological symbol recognizer is in the form of binary image planes. Each plane contains one or more symbols that fall into a particular shape category. The high-level module, which follows the symbol recognizer, is composed of two parts: a connected components analysis (CCA) module and a reasoning module. The CCA module performs connected component analysis on each bit plane output from the symbol recognizer. For each connected component in a bit plane, descriptors such as centroid, area, and bounding rectangle are computed. This information and the spatial relationships between the symbols are used to perform final recognition of the symbols.

In this section, we first describe the general postprocessing strategy and follow it up with specific details on processing a few important symbols.

The output from the feature extraction module is only a moderate partition of the symbols in the music score into several categories. This partition has to be refined by using prior knowledge about the elements in a music score. For example, the bit plane containing the staff lines has extraneous information that needs to be filtered out. Since the detection algorithm assumes that the staff line is a long horizontal line, horizontal lines that are part of long arc ties between notes are also detected during the detection step. Moreover, a single staff line may be broken into multiple pieces (because of tiny breaks in the input data). So further processing needs to be done to group broken lines and to filter out horizontal lines that are not part of the staff lines in the music score.

For each binary image, CCA is performed and we produce an intermediate data structure that consists of:

- Original image-id
- The component number
- The row and column coordinates of the centroid of the connected component
- The area of the connected component
- The upper left and bottom right coordinates of the rectangular bounding box of each connected component
- The parameters of the best-fitting ellipse for each connected component

The image-id identifies the preliminary category to which the component belongs. We chose the other descriptors such as centroid and bounding rectangle coordinates because these descriptors adequately encode the information in the image.

9.1 Algorithms for postprocessing and filtering

In this section we describe algorithms that we use during the postprocessing stage. The output obtained after postprocessing and filtering is an internal data structure encapsulating the information in the music score. This section describes some of the algorithms used after the feature extraction steps. We give specific details for extraction of staff lines, computation of note durations and note pitch values, and association of accidentals with notes. All subsequent processing is done by representing the symbol data internally in the form of linked lists. Several linked lists are formed from the CCA module output, including the bar and stem list, note list, clef list, and the accidental list. Subsequent processing is done by modifying/verifying the data in these linked lists.

Staff line processing. The binary image obtained as the output of the horizontal line extraction algorithm contains extraneous information that has to be filtered out. The detection step assumes that the staff line is a long horizontal line and the output is composed of a collection of long horizontal lines. As mentioned earlier, parts of arc ties across notes are horizontal segments and hence may be present in the output. Due to noise, a single staff line may get broken up into multiple horizontal pieces.

The staff lines are identified by using the following steps:

1. Detect long horizontal lines in the input image (feature extraction step).
2. Determine the centroid, area, and orientation of the connected components in the output from the feature extraction stage.
3. Determine disconnected horizontal segments that can be part of the same staff line. This is done by computing the perpendicular distances from the origin (top left point in the image) to the individual line segments. The perpendicular distance should be approximately the same for all the potential merger candidates.
4. Merge potential candidates if the distances between the adjacent endpoints of the line segments is less than a specified distance threshold.
5. Save the details for each staff line in a staff line array. This array is used to determine the pitch of each note.

The spacings between adjacent staff lines are then computed. This spacing gives the approximate sizes for the symbols expected in the music score. A robust estimate of the spacing is obtained by taking the median of all the spacings between the staff lines.

Distinguishing measure bars and note stems. Measure bars and note stems are both vertical lines. Measure bars are usually longer. For a music score that involves multiple voices, such as a piano sheet, we use the term measure bar to signify the vertical line that runs across both pairs of staffs (treble as well as bass). A significant difference between clefs and measure bars is due to their placement in the music score. Stems are always attached to note heads. The procedure used to remove stems from a list containing bars and stems does

this by intersecting each item on the list with the bounding rectangles of elements in the note list. All items that do not have any intersection are potential measure bars. Only measure bars that have significant enough area are retained as measure bars. Our current implementation stops at this step, but further checks could be made, such as checks for nearby clef symbols.

Note duration computation. The output from the feature extraction module consists of binary images containing filled blobs and unfilled blobs. In order to determine the duration of each note, a test for intersection of the bounding box for each notehead and some stem in the stem list was performed. If there is an intersection, then the note duration was halved. Then tests of intersection of the stem with beams, slants and squiggles are performed in order to identify quarter notes, single eighth notes, etc. The tests are performed in the order specified below:

1. Determine if note head intersects with some stem.
2. If there was an intersection with a stem, then determine if the stem intersects with some beam.
3. If there was an intersection with a beam, then determine if the beam intersects with some slant.

Association of accidentals with notes. Accidentals such as sharp, flat, and natural symbols are associated with notes or staff lines by using the following steps. Accidentals can be associated with notes when they occur next to a note. They can also be part of a key signature, in which case they occur next to a clef symbol. Because of this we use the following steps to associate accidentals with notes:

1. For each accidental in the accidental list, determine notes close to the accidental. Only notes that are in the bottom right quadrant, defined by coordinate axes with the origin as the upper-left corner of the accidental's bounding box, are potential candidates for association. This is due to the fact that the accidentals are always located to the left and above or on the same level as the notehead.
2. Compute the distance between the closest clef symbol and the accidental. If this distance is greater than the accidental-to-note-distance, then the accidental is associated with the note. If both the distances are greater than a specified fraction of the measure length (number of pixels between successive measure bars), then the current implementation flags the accidental as a possible misclassification. The clef-to-accidental distance is compared with the note-to-accidental distance because the accidental could be part of the key signature or be a modifier for the note.

10 Experiments and results

The printed score sheets we worked with were digitized at 300 dpi. The system runs on a MVI – Genesis 2000 image processing workstation and takes 2 minutes to process a 512×480 image. So far we have processed 74 images (512×480 pixels). There were 176 complete measures in all. There were 1311 occurrences of stock symbols (where a stock symbol is defined as a symbol that the system is designed to recognize) out of which 1252 symbols were recognized correctly. Thus the overall recognition accuracy is 95.5%. Figures 14 and 15 show the recognition results for one of the images with the recognized symbols overlaid. Table 1 shows the misdetection and misclassification rates for the different stock symbols. The misdetection and misclassification are defined as follows.

$$\text{Misdetection\%} = \left(1 - \frac{\text{number of correctly identified occurrences}}{\text{Total number of occurrences}}\right) \times 100$$

$$\text{Misclassification of symbol A} = \frac{\text{number of times a non-A symbol was recognized as A}}{\text{Total number of occurrences of non-A stock symbols}}$$

The misdetection was highest for the quarter rest and lowest for the quarter and eighth notes. This can be explained by the fact that the notes appeared with relatively less shape-size variation. In addition, the distinguishing features of notes are the note head, stem and the beams, all of which are stable features. For the quarter rest the bottom concavity was not a reliable feature. In a few cases, the concavity merged with the staff lines and was consequently obscured. The results can be improved if we include the foreground features of this symbol and exclude the bottom concavity from the detection sequence.

Table 1. Recognition results for the sample music score

Symbol type	Occurrences	Correctly identified (%)	Misclassification (%)
Half note	97	89 (91.75)	1 (0.0007)
Quarter and eigth notes	967	964 (99.7)	1 (0.0007)
Treble clef	24	23 (95.83)	0
Bass clef	23	21 (91.3)	0
Flat	63	61 (96.83)	4 (0.0031)
Sharp	13	13 (100)	0
Natural	23	22 (95.65)	0
Quarter rest	47	42 (89.36)	2 (0.0015)
Eigth rest	18	17 (94.44)	1 (0.0007)

There were in excess of 30 symbols in the scores we worked with, but the algorithms were designed to extract only around 12 of these symbols (symbols like grace notes, slurs, and note duration extension dots were not dealt with). This did not affect our results at all as is evidenced by the low misclassification rates.

11 Conclusion

In this paper we described the essential components of MUSER, a music score recognition system, and provided algorithms that are part of the system. The layout extraction

and symbol recognition modules utilize morphological operation sequences while the midlevel module utilizes conventional CCA methods. At present, our implementation of the algorithms for postprocessing is complete and we are in the process of implementing the procedures necessary to translate the linked list of music node structures into an ASCII format. For the data set used, the morphological symbol recognizer produced encouraging results, substantiating our earlier claim that mathematical morphology could be used as a powerful tool to analyze and detect shapes. The postprocessing steps employed reduce ambiguities that were present in the output. Postprocessing will significantly improve the performance of the system when the noise level in the input image is quite high. Additional music notation rules Roemer (1985) could be used as constraints in the reasoning step to further reduce ambiguity.

Acknowledgements. We would like to acknowledge Ms. Bei Wang, who helped us with the laboratory experiments.

References

Baumann S, Dengel A (1987) Transforming printed piano music into midi. SSPR 90(9):532–550

Costa M (1990) A practical guide to task-oriented sequences of morphological operations for use in image analysis. Technical Report EE-ISL-90-01, Department of Electrical Engineering, University of Washington

Fujinaga I (1988) Optical music recognition using projections. M.S. Thesis, McGill University, Faculty of Music, Montreal, Canada

Haralick RM, Sternberg SR, Zhuang X (1987) Image analysis mathematical morphology. IEEE Trans Pattern Analysis Machine Intelligence 9:532–550

Modayur BR (1991) Morphological algorithms for printed music score recognition. Technical report EE-ISL-91. Department of Electrical Engineering, University of Washington (In preparation)

Modayur BR, Haralick RM (1991) Music score recognition using mathematical morphology. In: Proceedings of the Fifth International Conference on Symbolic and Logical Computing. Madison, S.D., April 1991

Prerau DS (1970) Computer pattern recognition of standard engraved music notation. Ph.D. dissertation, MIT

Pruslin DH (1966) Automatic recognition of sheet music. Sc.D. dissertation, MIT

Read G (1969) Music notation – a manual of modern practice, 2nd ed. Allyn Bacon, Boston

Roach JW, Tatem JE (1988) Using domain knowledge in low-level visual processing to interpret handwritten music: an experiment. Pattern Recognition 21(1):33–44

Roemer C (1985) The art of music copying: the preparation of music and performance, 2nd edn. Roederick Music Company, Sherman Oaks