ISSUES IN THE DESIGN OF AN OPTICAL MUSIC RECOGNITION SYSTEM

Ichiro Fujinaga, Bo Alphonce, and Bruce Pennycook
Faculty of Music
McGill University
555 Sherbrooke St. W.,
Montreal, Quebec
CANADA H3A 1E3

The Optical Music Recognition Project at the Faculty of Music at McGill University in Montreal, Quebec has emerged from graduate research by Ichiro Fujinaga using the technique of projection to segment and analyze a wide variety of musical symbols from score. The OMR project has recently received funding from the Social Sciences and Humanities Research Council of Canada which has enabled the authors to port the preliminary system from DOS to a SUN 386i system. The system exploits the capabilities of NeWS, a graphics/windowing system much like PostScript. Various approaches to the problems of image enhancements, user interaction, music data structures for output storage, and automatic image learning systems are discussed. The paper will be accompanied by numerous slides showing various components and screen images of the system. A discussion of the current state of the OMR system will include performance benchmarks and an evaluation of conditions affecting the accuracy of the automatic recognition algorithms.

Background
In November, 1988, we presented to the Small Computers in the Arts Conference in Philadelphia the first progress report of the Optical Music Recognition project (Alphonce et al. 1988). This research, which is currently funded by the Social Sciences and Humanities Research Council of Canada, has evolved from a Master's Thesis by Fujinaga (Fujinaga 1988) wherein computationally efficient techniques for segmenting and identifying musical symbols from digitally scanned printed music were explored. The paper presents the current state of the OMR system and examines some of the central problems we have encountered. Examples of scanned, analyzed and re-constructed musical passages will be shown.

From DOS to UNIX
Since December, 1988, the system has undergone significant reprogramming on a SUN 386i (19" monochrome screen). The move from DOS to UNIX (SunOS 4.0.1) was dictated by the need for a better graphics display and programming environment. The PC-AT bus and DOS compatibility of the 386i has simplified the I/O process. A Datacopy 730 scanner (300dpi) is interfaced through an AT slot and controlled by a DOS program which produces TIFF format bit image files. Currently, these are converted to PostScript or SunView format prior to the display and analysis.

The user interface and graphics I/O are being developed using the SUN NeWS (Network Extensible Window System) imaging model, which is very similar to Display PostScript. We have adopted the NeWS standard because it is clear that numerous hardware and software vendors will be supporting NeWS (Atari ST, Apple A/UX, OS/2) thus providing the broadest possible range of low-cost target configurations for OMR. In fact, the program to display and
to print the recognized music is developed on the Mac IIx running A/UX 1.1 with the Grasshopper Group's NeWS.

Data Structure
Part of the mandate of the research project is to develop a comprehensive data format for musical information to simplify transmission of scanned and encoded scores among programs, systems, and sites. Well-known schemes such as MLSTRAN (Wenker 1970, 1974) or DARMIS (Erickson 1976) are being examined. However, these methods generate structurally "flat" lists (Gourlay 1986). At this point in the project, we are developing an OMR Interim File Format (OMIFF) which can be easily translated into other formats such as PostScript, MIDI, and LARMS. We hope to present the results of this aspect of our research next year. At this time, however, we will focus on the problem of processing scanned images of musical symbols.

Learning Mechanisms and Classification
Successful optical character recognition (OCR) systems offer built-in learning mechanisms. That is, alphabets of new types of fonts can be identified and added to the recognizer's database. The current state of the OMR system not only accepts music typographic fonts but will accept and learn totally new symbols, a crucial feature if the system is to deal with nonstandard notations. In order to implement this mechanism a classification scheme called the k-nearest neighbour (k-NN) rule is used.

Classification using the k-NN rule operates as follows: The system first compares the unknown sample x with each previously classified sample. If the majority of k (k ≥ 1) closest sample(s) to x belong(s) to class σ then x is also assigned to the class σ. "Closeness" or distance is determined by comparing a limited set of features (e.g. width, and area) of pairs of samples. It turns out that this simple rule has an error rate which is no worse than twice the minimal rate (Cover & Hart 1967).

Despite the intuitive appeal and the good error rate, use of the k-NN rule in practical applications has been frequently abandoned because of the storage and computational requirements. For example, assume that 200 different classes and 10 features (requiring 4 bytes of storage each) are used for classification and that each class is represented by 1000 samples in the classified set. Then 8Mb (200 × 10 × 4 × 1000) of storage is required, and, using a simple distance formula, a minimum of 2 million (200 × 10 × 1000) subtraction and 2 million comparison calculations are required for each sample to be classified.

The performance of the k-NN rule depends on the features selected, the number of samples, the type of distance measure used, and the size of k. It is clear that the number of calculations can be decreased if less features are used for the classification. The problem is to find the subset of features which will not degrade the accuracy. Unfortunately, it seems that the optimal solution to this problem for the k-NN classifier can be found only by trying every possible combination (Foroutan & Sklansky 1988). This results in a very large number of calculations requiring 2^N distance calculations, where f is the total number of features and N, the number of classified samples.

The number of samples can also be reduced by choosing a subset of classified samples which correctly identifies all of the classified samples. Furthermore, the classified samples can be restructured (e.g. a tree) so that most of the distance calculations may be eliminated. There are several distance measures available, such as Euclidean, City-block, square distance, Optimum-distance measure (Short & Fukunaga 1980) and Optimal-global NN-metric
(Fukunaga & Flick 1984). The problem, in general, is that the calculation costs increase with accuracy. The goal, therefore, is to find the simplest measure which does not degrade the accuracy.

It should be clear by now that in order to reduce the storage and calculation time of the NN classifier, a formidable number of calculations must be performed. Therefore we are introducing the concept of "sleep."

We observed that personal micro-computers are idle most of the time. The current system is designed in such a way that the computer "sleeps", that is, does something useful while the operator is away. (Of course, all time-sharing systems have hidden or background tasks which are performed in the absence of user intervention. The point here is that most personal systems are either idle or turned off when not in use.) The optimization of our classifier, therefore, can be performed during the "sleep" period. In another words, the self-improving calculations are undertaken "off-line."

The system as designed is a learning system in the sense that it improves its performance during the optimization (sleep) cycle. The optimizer is given a variety of methods to improve itself - then, whenever it has time, it tries out every possible combination of these methods to find and retain the best one.

Features
The features used by the present k-NN implementation include width, maximum height, area, rectangularity or rectangle fit factor, R, where

\[ R = \frac{A_0}{A_r} \]

where \( A_0 \) is the object's area and \( A_r \) is the area of its minimum enclosing rectangle; furthermore aspect ratio, \( A \), where

\[ A = \frac{W}{L}, \]

and \( W \) and \( L \) are the width and the length of the minimum enclosing rectangle, and, finally, "central moment" (Reeves et al. 1988). These features are extracted for the projection profiles (Fujinaga 1988) and their first and second derivatives. As described above, the system, not the user, determines which of these features are used in the classification process. It should be noted that adding other types of features is relatively simple and will eventually be explored.

Printing and User I/O
Since NeWS is very similar to PostScript, producing hard copy identical to screen images presented on the screen is trivial. This was the primary motivation for our selecting NeWS as the basis for all I/O development and as the imaging model. Another important practical consideration is that Adobe'sSongeFonthas eliminated the immediate need to design a local set of music fonts.

OMP presents the user with a "split screen" (actually two windows) — one of which displays the original bit-image, the other the "recognized" score. On-line editing tools permit the user to correct interactively the recognized version. User intervention is essential for these kinds of applications in that no system (including most humans) can be expected to recognize all musical symbols with 100% accuracy.
Work in progress

Most of the current effort is directed towards implementing the various optimization techniques for the k-NN classifier. Given an "empty" system with, initially, intense user-intervention, and many hours of productive "sleep", an efficient recognizer can be automatically developed. We have begun intensive development (under ARI and VeWS) of the user-interface and specification of a music symbol data structure. Our current musical target is simple, polyphonic music (Bach chorales), where there is only one note per stem. Our next target is traditional piano music.

References


