Introducing Feedback into an Optical Music Recognition System

John R. McPherson
Department of Computer Science
University of Waikato
Hamilton
New Zealand
jrm21@cs.waikato.ac.nz

ABSTRACT

Optical Music Recognition is the process of converting a graphical representation of music (such as sheet music) into a symbolic format (for example, a format that is understood by music software). Music notation is rich in structural information, and the relative positions of objects can often help to identify them. When objects are unidentified or mis-identified, many current systems "coerce" the set of objects into some semantic representation, for example by modifying the detected durations. This could cause correctly identified symbols to be modified. The knowledge that the current set of identified symbols cannot be semantically parsed could instead be used to re-examine some of the symbols before deciding whether or not the classification is correct. This paper describes work in progress involving the use of feedback between the various phases of the optical music recognition process to automatically correct mistakes, such as symbolic classification errors or mis-detected staff systems.

1. INTRODUCTION

Optical Music Recognition (OMR) is of great importance to music information retrieval — there is an enormous amount of music that is currently only available in printed form. The process is typically split into distinct phases, such as staff location and/or pre-processing, object location and identification, and musical semantics analysis. The work described in this paper is a continuation of the work by Bainbridge on extensible optical music recognition [1]. The result of that work, the CANTOR system, is highly customisable, using configurable components such as pattern descriptions, grammars, and sets of rules to recognise staff-based music notation, including common music notation and plainsong notation.

Until recently, most of the work in OMR involved making each stage of the feed-forward system as robust as possible, allowing for mistakes by previous stages. Fujinaga and Droettboom [2] describe a system with two distinct subsystems: a glyph recognition stage and a semantic interpretation stage. Seales and Rajasekar [3] describe a rule-based system using feedback to do further processing on noteheads and stems that cannot be semantically joined. Baumann [4] uses an attributed graph grammar to perform primitive assembly and semantics analysis. He mentions problems due to the quality of the primitive recognition, and suggests the possible benefit of feedback from the grammar to the symbol recognition phase. Stückelburg and Doermann [5] describe the use of stochastic methods for finding the semantics that best describes the input image.

By using feedback, additional context can be used to determine how to automatically correct errors, rather than trying to arbitrarily modify an invalid semantic structure that was created by a single pass. For example, a common tactic for dealing with bars with an incorrect duration (relative to the time signature) is to insert rests. In the system under development, the coordinating process has no knowledge at all about music (either semantics or primitives). Its task is to determine the order of execution of the specialist modules, and handle requests generated by them. Figure 1 shows how the flow of execution is no longer a static, linear path. No part of the system assumes that a request will be accepted. If the coordinator is set to reject all requests, the system runs in a typical "feed-forward" manner. Most of the feedback is expected to be from the assembly and semantics stages to the pattern recognition stage, although the semantics stage will also send feedback to the assembly stage. For example, the pattern recognition stage could be told that a particular object is incorrectly classified, and be given a list of possible types that would make sense semantically based on the nearby identified objects.

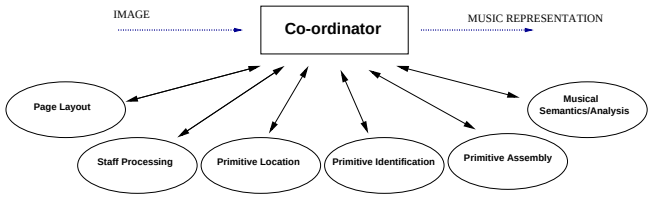

Figure 1: Framework showing sample 'specialist' modules

2. PATTERN RECOGNITION

The pattern recognition module is the main recipient of feedback when the current set of identified objects cannot be perfectly explained by the semantic rulesets.

Possible methods for this module to take account of any new information include:

- Storing more than one possible classification (and a matching certainty rating) for each graphical object. This means later stages could try different arrangements of objects based on these classifications. The "best" arrangement could be used to increase the certainty for a particular classification of this object.
- Loosening restrictions on certain patterns. For example, if we can determine that an object could be an accidental (for example, a sharp or flat) based on the objects around it, the system might lower the tolerances on the patterns for accidentals.
- Trying alternate patterns. We should be able to have a series of patterns that describe a treble clef (based on different publishers' fonts, for example), which are either manually or automatically (or randomly) given an order to test in. This could be done by keeping a history of which is the most successful pattern, or optically recognising the publisher's name.
- Using a different methodology for recognition, such as a neural network or Fujinaga's adaptive system.

3. FLEXIBILITY OF DESIGN

The modular design makes it easier to add functionality to the system. An example is that a module designed to correct segmentation errors was easily created in isolation, without major modifications to the other modules. The only modifications required were:

- the coordinator had to include the new module, and execute it after the primitive identification modules; and
- the primitive identification module and the new module had to agree on the name of the request used to call the former again, to process any changes made by the new module.

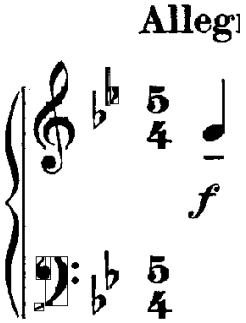

Figure 2: Broken Objects as a Result of Staff Processing

Figure 2 shows the start of the first staff from "Promenade" by Mussorgsky, after the Staff Processing module has removed the staff lines. This process can fragment objects — bass clefs and flats are particularly prone to this as they contain thin segments that coincide with staff lines. In this figure, the three fragments of the bass clef and the two fragments of the top-most flat are not identified with the default set of pattern descriptions. The new Segmentation specialist joins the nearby objects together into conglomerate objects and sends a request to the Primitive Identification specialist to process any newly created objects. This results in both objects being correctly identified.

Another advantage of the design is that the current primitive recognition stage could be easily replaced or augmented by another pattern recognition scheme as mentioned previously, such as Fujinaga's adaptive recognition.

4. CURRENT PROTOTYPE

Symbolic classification is achieved by creating physical descriptions of primitives. This allows arbitrary notations to be recognised — as a proof-of-concept, description files were written to recognise chess pieces from a computer-generated chess diagram, although no semantic rules were written to describe the layout of the chess board. The prototype (at the time of writing) made limited use of feedback. For example, feedback is used by the staff processing module to double check systems if skew is detected, or to use more expensive algorithms if no staff systems were found. At the time of writing, the prototype only had basic semantics support for common music notation. Feedback from the semantics module was not successfully used by the primitive identification stage to modify object classifications. The only musical file format currently supported for output is the GUIDO file format, although files in PostScript format can be produced to show the internal state of the lattice.

5. DISCUSSION AND FUTURE WORK

As previously mentioned, the coordinating module does not contain any information about music structure or graphical objects. This design ensures that the system will not be limited to any particular music notation. Insight may also be gained into the applicability of these techniques to non-musical recognition domains. One problem facing researchers in recognition domains is how to represent semantic information. No ruleset is able to completely describe something as ill-defined as common music notation, as composers often break with convention where convenient [6]. An important part of OMR is text recognition, although experimentation with free optical character recognition systems has shown mixed results. However, because so much information on a musical score is text — for example, performance notes, document metadata, and instrument names — a text recognition module must be considered.

A good overview of diagram analysis in general is given by Blostein et al. [7]. Of particular relevance is discussion of blackboard systems and contextual feedback. Different coordinating strategies will be developed and evaluated — factors such as overall accuracy, level of user intervention required, and processor and memory resources are of interest. However, as noted by Bainbridge, comparisons between different recognition systems are problematic due to both differing representations or formats used by the systems and the difficulty in determining relative 'correctness' of recognised scores.

Future versions of the system might have two or more pattern recognition modules, and use coordinating strategies that take advantage of this. For example, a voting mechanism could be used to classify objects, or the coordinator could randomly choose which module to use first, or preference could be given to the module that has historically made the least detected classification errors. Other coordinating strategies include blackboard systems and frame-based systems.

6. REFERENCES

[1] David Bainbridge. *Extensible Optical Music Recognition*. PhD thesis, University of Canterbury, Christchurch, New Zealand, 1997.

[2] Michael Droettboom. Selected research in computer music. Master's thesis, The Peabody Institute of the John Hopkins University, Baltimore, Maryland, USA, April 2002.

[3] W. B. Seales and A. Rajasekar. Interpreting music manuscripts: A logic-based, object-oriented approach. In R. T. Chin, H. H. Ip, A. C. Naiman, and T.-C. Pong, editors, *Image Analysis Applications and Computer Graphics, Third International Computer Science Conference*, pages 181–188, Hong Kong, 11–13 December 1995. Springer.

[4] S. Baumann. A simplified attributed graph grammar for high-level music recognition. In *Proceedings of the Third International Conference on Document Analysis and Recognition*, ICDAR '95, pages 1080–1083, Montréal, Canada, August 1995. IEEE.

[5] Marc Vuilleumier Stückelberg and David Doermann. On musical score recognition using probabilistic reasoning. In *Proceedings of the Fifth International Conference on Document Analysis and Recognition*, ICDAR '99, Bangalore, India, 1999. IEEE.

[6] Donald Byrd. Music notation software and intelligence. *Computer Music Journal*, 18(1):17–20, 1994.

[7] D. Blostein, E. Lank, and R. Zanibbi. Treatment of diagrams in document image analysis. In M. Anderson, P. Cheng, and V. Haarslev, editors, *Theory and Applications of Diagrams*, volume 1889 of *Lecture Notes in Computer Science*, pages 330–344. Springer Verlag, 2000.