AUTOMATIC PITCH RECOGNITION IN PRINTED SQUARE-NOTE NOTATION

Gabriel Vigliensoni, John Ashley Burgoyne, Andrew Hankinson, and Ichiro Fujinaga
Centre for Interdisciplinary Research in Music Media and Technology (CIRMMT)
McGill University, Montréal, Québec, Canada
[gabriel,ashley,ich]@music.mcgill.ca, andrew.hankinson@mail.mcgill.ca

ABSTRACT

In this paper we present our research in the development of a pitch-finding system to extract the pitches of neumes—some of the oldest representations of pitch in Western music—from the *Liber Usualis*, a well-known compendium of plainchant as used in the Roman Catholic church. Considerations regarding the staff position, staff removal, *space-* and *line-zones*, as well as how we treat specific neume classes and modifiers are covered. This type of notation presents a challenge for traditional optical music recognition (OMR) systems because individual note pitches are indivisible from the larger ligature group that forms the neume. We have created a dataset of correctly-notated transcribed chant for comparing the performance of different variants of our pitch-finding system. The best result showed a recognition rate of 97% tested with more than 2000 neumes.

1. INTRODUCTION

Optical music recognition (OMR) is the process of turning musical notation represented in a digital image into a computer-manipulable symbolic notation format. Music notation, however, comes in many different forms, and so there is no single OMR system that can recognize all types of music.

Plainchant is a large collection of monophonic melodies, which exist as one of the oldest types of notated music in Europe. These melodies have been part of (Western) Christian liturgy since the Middle Ages and have formed the basis for much Western music that followed. Its unique system of notation, constructed on groups of pitches known as *neumes*, is still in use in liturgical settings and was most famously re-popularized in the late 19th Century by the monks at the Solesmes monastery in France.

Perhaps the most used book produced by the Solesmes community was the *Liber Usualis*, a 2000-page service book that contains most of the texts and chants for the offices and masses of the church, designed to be a practical reference book by those responsible for performing these services. As a modern production, many qualities of older manuscript sources—notably the variation in scribes handwriting and degradation due to age—are not present in the *Liber*, but its sheer size and comprehensiveness provides an excellent foundation for training automatic neume recognizers.

In this paper we present our work to date for producing a neume recognition system. This work concerns the recognition of printed neume pitches, specifically in the style used throughout the *Liber*, known as *square-note notation*. In square-note notation, most neumes are ligatures that represent multiple pitches, and so accessing the individual notes of a neume can be problematic. We have developed a system whereby we recognize the position of the first pitch of a neume group, and then employ a unique approach using automatic class labels to recognize the remainder of the pitches in that group.

1.1 Notation: From Neumes to MEI

We have chosen the Music Encoding Initiative (MEI) format[1] as the basis for capturing the output of our recognition system. This format provides an extensible approach to encoding different types of music documents [7]. As part of this project, we have developed an extension to the MEI format that captures the particular qualities and nuance of Solesmes-style square-note notation. We convert the output from the recognition system into MEI files through the use of the PyMEI library[2].

1.2 Notation Systems

Plainchant notation is a precursor to modern music notation. Initially it was conceived as a means of providing some indication of melodic contour to a text that was chanted during

[1] http://music-encoding.org/
[2] https://github.com/ahankinson/pymei

a liturgical service, and provided no indication of absolute pitch values. With the invention of the musical staff between the 9th and 10th Centuries, some forms of neumes made the transition to being "heightened," or given absolute pitches in relation to a staff and clef. Solesmes notation features absolute pitch, and so its pitch values may be extracted directly.

Dalitz et al. [1] have presented a system for symbol recognition of Byzantine chant notation using neume forms unique to this repertoire. This system of notation, however, does not use a staff to specify absolute pitch. The neume shapes for this type of notation specified melodic contour and relative pitch direction, and the authors do not report any attempt at supplying absolute pitch information. Gezerlis and Theodoridis [2] have also presented a system for recognizing Byzantine chant notation, however, like Dalitz et al., their system does not perform any pitch transcription.

Laskov and Dimitrov [3] have presented a system for performing image segmentation for neume notation, separating the neumes as objects of interest from the background. Again, however, this system is for unpitched neumes and thus the authors made no attempt at extracting pitches.

Finally, Ramirez and Ohya [4] have presented a classification system used to classify eight basic types of neumes in pitched manuscript sources from the 14th Century. This system classifies these symbols accurately, but the authors did not report any attempt at performing pitch recognition from the recognized classes. The authors also made no indication of how their system would perform outside of the eight standard neume forms they used.

In our paper, we address a method of performing neume classification and pitch transcription from 11 basic types of neumes and its multiple variants presented on a staff.

2. WORKFLOW

We are creating a fully searchable, web-based version of the *Liber*. This project will allow users to search for specific melodic sequences or patterns in the book; our system will return all matching patterns, highlighting them in the original image. To accomplish this goal, we needed to extract all musical and textual information from the *Liber*. Although it contains text and music, this paper will focus only on the music. We are using the digitized, downloadable version of the *Liber* published by Desclée & Co in 1961 [3], which comes as a PDF file with 2340 pages. Different categories of information and content can be found on the pages: title, text, staves, *lettrines*, and lyrics. These were separated into different layers during an automated preprocessing step to allow for easier content analysis.

2.1 Page Preprocessing

We converted the PDF file into Tagged Image File Format (TIFF) image files so that they could be read by *Aruspix* [5], a cross-platform application devoted to optical music recognition on music printed during the European Renaissance.

The preprocessing capabilities of Aruspix include skew correction, resizing, cleaning, staff-position retrieval, and classification of the elements in a page in the following categories: frames and borders, *lettrines*, lyrics, inter-staff elements, and titles. These page elements are uniquely coloured to allow the separation of the page elements into discrete layers for extraction. The colouring feature is also useful for manual correction. Aruspix returns a container file, which includes the original binarised TIFF image, a second coloured TIFF image with each one of the page elements in a different colour, and an XML file with information about the processes performed on the page. After automatically preprocessing the entire TIFF images using Aruspix, we manually corrected any misclassified page elements. We then modified the *Gamera4Aruspix* (G4A) Gamera Toolkit [4] to extract the preprocessed layers into separate images, providing us with images containing only music staves.

2.2 Retrieving Staff Position and Removing Staff-Lines

We processed the images containing only music notation with the MusicStaves Gamera Toolkit [5]. This toolkit allowed us to extract each staff line position in the staff, store its location for later determination of pitches and remove all lines for later glyph classification. MusicStaves is designed to work with an arbitrary number of lines, which was required because the *Liber* features 4-line staves.

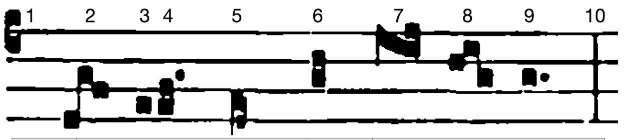
2.2.1 Staff Detection

The MusicStaves toolkit has a number of different algorithms for detecting the position of staff lines in the image and removing them, leaving only the remaining elements in the resulting image. These algorithms provide different results depending on the notation style and image deformations. We tested two different staff-finding algorithms. For the first approach, we retrieved the average vertical position of each one of the lines horizontally across the whole page, and named this approach *AvgLines*. This approach allowed us to determine the gross slope of a staff line across the page and thus to correct for staff lines that are not straight. In the second approach, we used the Miyao algorithm, capable of breaking a staff line into equidistant segments to provided a more precise means of determining horizontal staff slope

[3] `http://musicasacra.com/2007/07/17/liber-usualis-online/`

[4] `https://github.com/vigliensoni/G4A`

[5] `http://lionel.kr.hs-niederrhein.de/~dalitz/data/projekte/stafflines/`



Glyph	Labeling	Glyph	Labeling
1	clef.c	6	neume.podatus.2
2	neume.torculus.4.3	7	neume.porrectus.2.2
3	neume.punctum	8	neume.torculus.2.3
4	neume.podatus.2.dot	9	neume.punctum.dot
5	neume.cephalicus.2	10	division.major

Figure 1. Extract of neumes in page 1045, stave 2 of the *Liber* with the notation we developed for encoding the glyphs and intervals for each one of the neume variations.

changes in inclined, braked, or flexed staves [8]. We preserved the position of all *staff segments* and named this approach *Miyao*.

2.2.2 Staff Removal

A previous study by Dalitz et al. [6] shows that there is no single superior algorithm for performing staff removal. On a number of different metrics, they observe that for images with deformations, the performance of many algorithms for staff removal is very similar, and so there is no clear best technique. We informally tested the five algorithms provided by MusicStaves and determined that for our repertoire, the Roach & Tatem algorithm performed the best.

2.3 Neume Classification and Labeling

After the detection and removal of the staves, the images were loaded into the Gamera interactive classifier. The glyphs from 40 pages of the *Liber* were manually classified to create a training dataset for later automatic classification. Our pitch-finding approach would only retrieve the position of the first part of a glyph, so for different variations of neumes, an encoding scheme was developed that uniquely captured the shape of the entire neume, grouping like neumes into the same class. This encoding scheme identified the intervals as well as other auxiliary shapes in the neume and served as a class identifier. Those other elements that have an effect on the shape of a neume, such as dots, horizontal and vertical *episemas*, the *quilismas*, which are alterations of the note shapes, or the combination of some of those elements, are explicitly declared and encoded. Figure 1 shows an excerpt of the *Liber* with the notation we developed for encoding the neumes and other notational elements. It can be seen that the interval between two consecutive points in a neume is encapsulated into the glyph class name in the Gamera interactive classifier. Consequently, finding the pitch of the starting note allows us to derive pitches for all notes in a given neume.

2.3.1 Automated Neume Pre-classification

After manually training the classifier with 40 pages of neumes, we had a dataset large enough to be used as a model for automatic classification of elements for the remaining pages. A classification and optimization script was developed to automatize the process of classifying elements in the new pages and optimize the classifier. By this means, the classifier increased in size with new neumes or new neume variations only.

2.3.2 Human-supervised Neume Classification

There were inevitable errors in the automatic classification, and so we performed a manual correction of each page by musicologists trained in the Solesmes notation system. This ensured that all glyphs for every single page were correct.

3. PITCH-FINDING APPROACHES

To calculate the pitch of the starting point for each neume, we needed to know its location on the staff as well as the clef type and its position. To accomplish this, we created imaginary lines, *ledger lines*, and zones, *line-* and *space-zones*, in the staff and calculated the placement of neumes in these zones.

3.1 Ledger Lines

We detected the staff-line positions using the Miyao and AvgLines approaches described previously. However, some neumes, or notes inside a neume, were located on ledger lines above the first (upper) or below the fourth (lower) line. Therefore, we virtually extend the number of staff lines by creating four imaginary ledger lines, two above the stave and two below. The distance between two staff lines for the ledger lines was determined by projecting the distance of the closest actual staff lines as the ledger lines.

3.2 Space- and Line-Zones

Notes of a given neume can be located either on the staff lines or in the spaces between lines. We defined imaginary zones between the staff lines where the neumes could be located. We calculated these imaginary zones by segmenting the space between two lines into four segments. The second and third segments correspond to a *space-zone*, and the first and fourth were grouped with the previous fourth and first, respectively, to designate a *line-zone*. We assigned to each zone an unique number which corresponds to what we named the *note-position* of each neume on the staff. Figure 2 shows the imaginary ledger lines and zones in a stave.

3.3 Bounding Box

To determine the pitches of the neumes, we first tested an approach based on the bounding boxes of each one of the

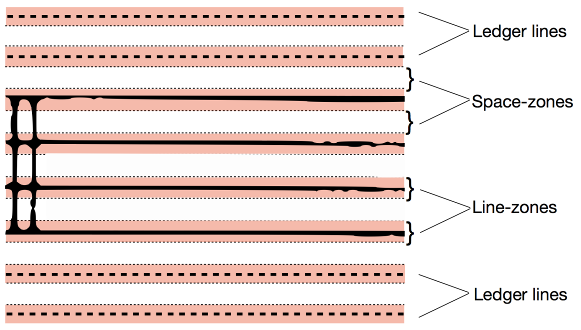


Figure 2. Stave showing imaginary upper and lower *ledger lines* as well as *line-* and *space-zones*.

glyphs—that is, the rectangular area that defines the bounds of a particular glyph. These bounding boxes are generated by Gamera and can be easily accessed. We determined that many neumes began in either the top left of the bounding box (the *up* position) or the bottom left of the bounding box (the *down* position). A neume position was pre-determined for its specific neume class, and the up or down position of the first note was correlated with all possible line or space zones in a staff. From there the pitch of the first note of the neume was determined. Although this approach worked for most of the neumes, some of them, especially *torculus* and *compound* neumes (see Figure 1, glyph 8), do not necessarily start with a note in the upper or lower position. The starting note of those neumes can be located between their bounding box's left vertices, making the bounding box approach impractical for use across the entire *Liber*. Because of this limitation we abandoned this approach.

3.4 Horizontal Projection and Center of Mass

A more robust approach is based on the horizontal projection of the neumes. To find the starting note for each one of the neumes we created a sub-image—a vertically split version of the original neume with a width of the size of a single *punctum*. This unit was chosen because we considered the punctum (see Figure 1, glyph 3) as the nominal neume unit. Before creating the sub-images, we calculated the average punctum size among all punctums on a page and used that size for any sub-image creation. We then retrieved the horizontal projection of each one of the neumes and calculated its centre-of-mass, that is, the point around which the black pixels of the sub-image were equally distributed. By using this method, we found the mean location of the starting position of a neume, and we determined the staff zone for this starting point, allowing us to automatically derive its starting pitch.

3.5 Clef Type and Position for Shifting Pitch Notes

The pitches of the neumes in a staff depend on the clef type, *C* or *F*, and its line position on the staff. The Gamera classification process does not treat the clef in a special way, and so a method must be devised to allow any pitch-finding system to automatically detect the closest clef on a given staff so that all notes on that staff may be correctly identified. The coordinates (relative to the staff boundaries) where each neume was located was stored temporarily. After all elements in the staff were classified, they were then sorted according to their vertical position, rounded to the nearest staff boundary, and then by their position on the x-axis, left to right. This produced a sequence of neumes, ordered by staff and then occurrence on the staff, and their pitch was assigned according to the clef, which is always the first element for each stave.

3.6 Special Neumes and Neume Modifiers

Although the centre-of-mass approach provided a more robust method of initial pitch detection, there were still some neume shapes that required further processing, in particular the *podatus*, *epiphonus*, *cephalicus*, *scandicus*, and their variations (see Figure 1, glyphs 4, 5, 6, and 7). These shapes have sub-images with two notes or elements that are vertically stacked, shifting the centre-of-mass and making our projection system an inaccurate method of finding the initial pitch position of the neume on the staff. To fix this issue and improve accuracy, we made a number of exceptions where the sub-image was determined by first splitting these neumes horizontally. The centre-of-mass of the sub-image's largest connected component was then considered the centre-of-mass of the neume. Similar processing was needed for neumes that had horizontal episemas, vertical episemas, or dots. Their centre-of-mass was shifted due to the presence of these modifiers, and so we removed these elements before calculating their vertical position on the staff. We left this feature of treating some neumes and neume modifiers in the described way as an option for testing its performance in comparison to the standard approach. We named the former *Exceptions* and the latter *No Exceptions*.

3.7 Moving the Space- and Line-Zones

The position of the space- and line-zones in relation to the staff lines has an impact in the performance of the pitch-finding system, and we informally tested several values for this relation in order to see how its performance could be improved. For the final comparison of settings, we tried two approaches for the spacing: a regular spacing of the zones, already described, and a shifted spacing, with the upper line of the space-zone shifted down by 2/16 and the lower one by 1/16 of the staff-space in that staff segment.

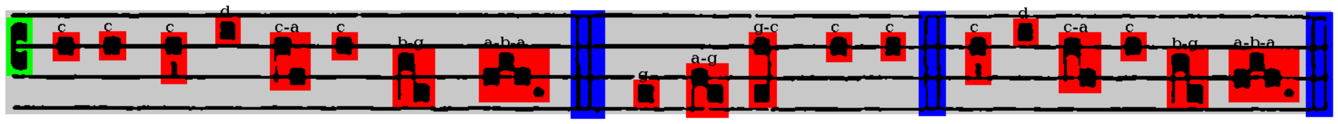


Figure 3. Visualization of pitch find algorithm performance in page 1242 stave 5 of the *Liber Usualis* using the Python Imaging Library and the original image.

4. TESTING AND RESULTS

To evaluate the performance of our six pitch-finding systems and their variants (Miyao, AvgLines, Exceptions, No Exceptions, Regular Spacing, Shifted Spacing), we created a ground truth dataset consisting of 20 random pages with a total of 2219 neumes and 3114 pitches correctly labeled. We used the MEI format for storing the music notation. This format has the ability to correlate zones on an image with musical structures encoded in XML [7]. We developed a script for highlighting neumes on the page image and identifying its pitch in text next to the note on the screen (Figure 3). This visualization tool allowed us to quickly identify and correct the miscalculated pitches in the MEI file in order to create a ground truth dataset.

We tested our system using six different variants based on the staff-line detection approaches we developed, the treatment of glyphs with special conditions, and a shift in the spacing of the line- and space-zones. The nomenclature we used for these variants can be seen in Table 1.

Nomenclature	StaffLine detection	Exceptions	Zone Spacing
ANR	AvgLines	No	Regular
AER	AvgLines	Yes	Regular
MNS	Miyao	No	Shifted
MNR	Miyao	No	Shifted
MER	Miyao	Yes	Regular
MES	Miyao	Yes	Shifted

Table 1. Nomenclature used for the different variants of the experimental testing for the performance of our pitch-finding system.

Figure 4 shows the recognition rates achieved by the six methods for the different neume classes (the *compound* and *scandicus* neumes are not included in the graph because they are relatively rare in our dataset.)

Overall, the best variant performance was MES, i.e., the variant that included the Miyao algorithm, the handling of special exceptions, and correction for vertical spacing, with a 97% recognition rate in finding the pitch of the first note of a neume only. This value was reduced to 95% when we retrieved and compared all notes from all neumes with the ground-truth dataset. ANR performance was the worst at 85% for the first note pitch and 81% for all pitches.

From the graph we can see that the *cephalicus* and *podatus*, and to a lesser degree the *epiphonus*, *scandicus*, and *torculus*, share a common pattern: their best results were achieved with the MES, AES, and MER variants, i.e., the variants that include handling of the special exceptions. These are the neume shapes for which the special exceptions to the centre-of-mass approach were designed, and so, it is clear that these special exceptions were necessary. On the other hand, *clivis*, *porrectus*, and *punctums*, and to a lesser degree, the *virgas*, have in common that their high performance was accomplished using MES and MNS, i.e., the variants that include both the Miyao algorithm and correction for vertical spacing. Collectively, these neumes represent more than 80% of all neumes in the dataset, and so it is clear that the use of the Miyao approach with shifted spacing is important for achieving the most accurate recognition possible. Confirmatory testing with logistic regression showed that overall, all three of our innovations—use of the Miyao algorithm, special treatment of exceptions, and spacing correction—produced statistically significant (α = 0.05) improvements to the recognition rate. Furthermore, as Figure 4 illustrates, these improvements are large enough to make an important difference in the quality of our output, bringing the overall recognition rate from 85% to 97%.

5. FUTURE WORK

Formal research should be done to determine the best position and spacing of the line- and space-zones. We discovered that this feature is an important factor to find the correct pitches for some neumes. Secondly, we want to calculate the performance of all the automated workflow, including the neume classification and pitch recognition, to see how our system performs automatically, without human supervision.

As was stated, at the end of our project we will have the entire *Liber Usualis* fully transcribed and searchable. We hope that it will be a great source of information for musicologists as well as church goers and musicians. At the same time, however, we will have a massive ground truth of correctly transcribed melodies and neumes, and it could be used as the starting point for digitizing and research using other books and manuscripts with Solesmes and similar notation.

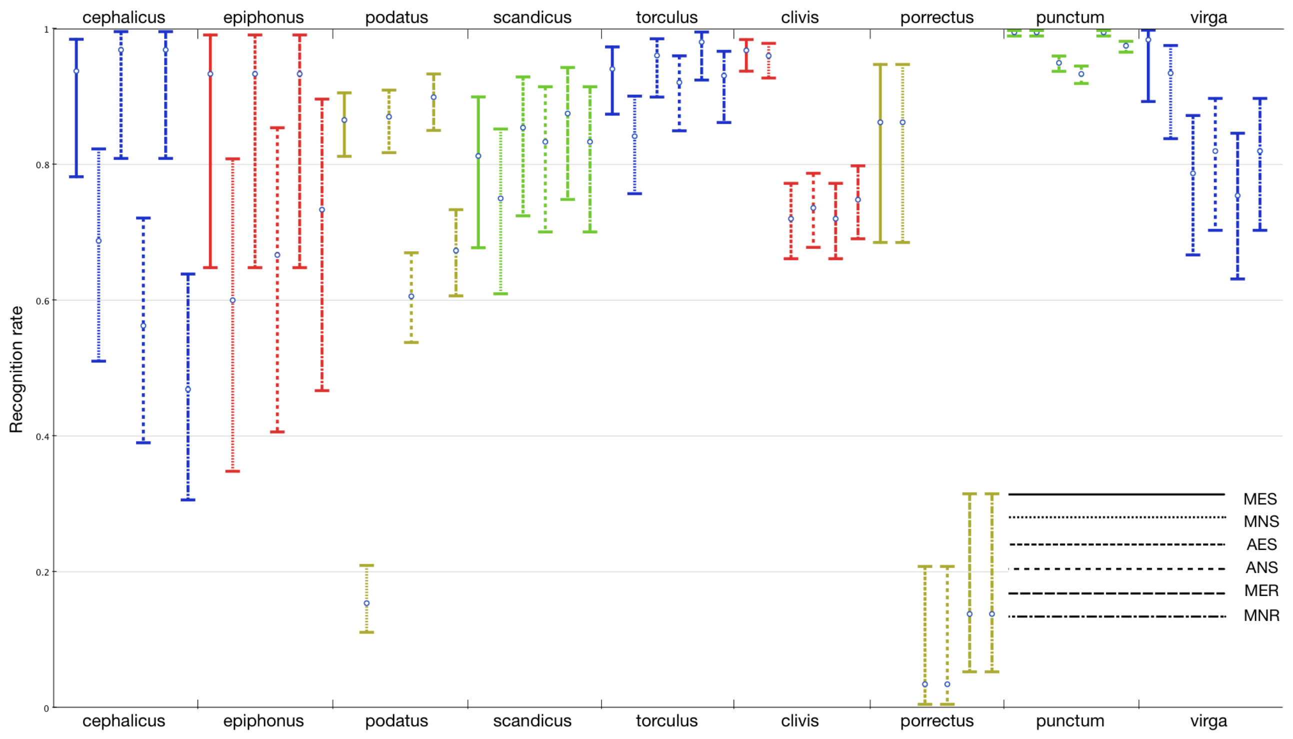


Figure 4. Precision and error bars for the neume classes in the ground truth dataset in finding pitches of the six variants we tested.

6. ACKNOWLEDGEMENTS

The authors would like to thank our great development team for their hard work: Remi Chiu, Mahtab Ghamsari, Jamie Klassen, Saining Li, Wendy Liu, Mikaela Miller, Laura Osterlund, Alastair Porter, Laurent Pugin, Caylin Smith and Jessica Thompson. This project has been funded with the generous financial support of the *Social Sciences and Humanities Research Council* (SSHRC) of Canada.

7. REFERENCES

[1] Dalitz, C., G. Michalakis, and C. Pranzas. 2008. Optical recognition of psaltic Byzantine chant notation. *International Journal on Document Analysis and Recognition* 11 (3): 143–58.

[2] Gezerlis, V. G., and S. Theodoridis. 2002. Optical character recognition of the orthodox Hellenic Byzantine music notation. *Pattern Recognition* 35 (4): 895–914.

[3] Laskov, L., and D. Dimov. 2007. Color image segmentation for neume note recognition. *Proceedings of the International Conference on Automatics and Informatics*. Sofia. 37–41.

[4] Ramirez, C., and J. Ohya. 2010. Symbol classification approach for OMR of square notation manuscripts. *Proceedings of the 11th International Society for Music Information Retrieval Conference*. Utrecht. 549–53.

[5] L. Pugin. 2009. Editing Renaissance Music: The Aruspix Project. In *Digitale Edition zwischen Experiment und Standardisierung Musik - Text - Codierung*, ed. Stadler, P., and J. Veit. 147–56. Tübingen: Max Niemeyer Verlag.

[6] Dalitz, C., M. Droettboom, B. Czerwinski, and I. Fujinaga. 2008. A comparative study of staff removal algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30 (5): 753–66.

[7] Hankinson, A., L. Pugin, and I. Fujinaga. 2010. An interchange format for optical music recognition applications. *Proceedings of the 11th International Society for Music Information Retrieval Conference*. Utrecht. 51–6.

[8] Miyao, H., and M. Okamoto. 2004. Stave extraction for printed music scores using DP matching. In *Journal of Advanced Computational Intelligence and Intelligent Informatics* 8 (2): 208–15.