Creating a Large-Scale Searchable Digital Collection from Printed Music Materials

Andrew Hankinson
andrew.hankinson@mail.mcgill.ca

John Ashley Burgoyne
john.ashley.burgoyne@mail.mcgill.ca

Gabriel Viglianesoni
gabriel@music.mcgill.ca

Ichiro Fujinaga
ich@music.mcgill.ca

Centre for Interdisciplinary Research in Music Media and Technology (CIRMMT)
Schulich School of Music, McGill University, Montréal, QC, Canada

ABSTRACT

In this paper we present our work towards developing a large-scale web application for digitizing, recognizing (via optical music recognition), correcting, displaying, and searching printed music texts. We present the results of a recently completed prototype implementation of our workflow process, from document capture to presentation on the web. We discuss a number of lessons learned from this prototype. Finally, we present some open-source Web 2.0 tools developed to provide essential infrastructure components for making searchable printed music collections available online. Our hope is that these experiences and tools will help in creating next-generation globally accessible digital music libraries.

Categories and Subject Descriptors

H.5.5 [**Sound and Music Computing**]: Systems

Keywords

Optical music recognition, music notation, music score searching, web applications.

1. INTRODUCTION[1]

Document digitization is now operating at an industrial scale within libraries and archives. Every year, millions of books are digitized and placed online. Increasingly sophisticated methods of searching, browsing, analyzing, and retrieving the textual content of these books are being developed to allow users to navigate these texts. Initiatives like Google Books [1], the HathiTrust [2], and OpenLibrary [3] have been digitizing, recognizing, and indexing large amounts of textual material. The IMPACT project [4] has focused exclusively on developing tools, technologies and best practices for analyzing historical texts, further advancing the state of the art of computational tools that work with older, degraded, or complex layouts and typefaces. Optical character recognition (OCR) is the central technology in all of these projects, transforming page images into text that can then be stored and indexed. No similar initiatives for musical materials exist at this scale and scope. Most optical music recognition (OMR) software is not designed to process large volumes of page images. Digital music document libraries still rely on human-supplied metadata (book titles, for example) as the primary means of navigating large document image collections. The same modes of search and navigation that are available in the large textual digitization projects—allow users to search every book in a collection at the page level in a matter of seconds—are unavailable for musical materials.

To address this discrepancy, we introduce the Single Interface for Music Score Searching and Analysis (SIMSSA) project [5], a recently funded initiative to investigate the creation of a fully searchable digital musical document collection, including document digitization, high-volume OMR, large-scale symbolic search systems, and digital document display. Here, we report the results of our initial investigations for scaling musical document processing to accommodate large volumes of page images as well as some of the techniques and software tools we have developed to meet the unique challenges this process presents. Finally we discuss future directions for the SIMSSA project and how we think this project will change the ways people interact with digital musical documents.

2. BACKGROUND

Prior to the large-scale digital document initiatives, both OCR and OMR technologies were primarily used as a method of document transcription: A page image was supplied, the words or music notation was extracted, and the original image was then discarded. The user was left with the content of the page in an editable format that could be opened in word processing or notation editing software. Automatic transcription of page images was an alternative to entering the content into the computer by hand, an arduous task for documents of any appreciable length.

With book digitization projects, however, the use of OCR changed. It was no longer used primarily as a means of making a text editable, but as a means of making a collection of images navigable. This shift meant that it became very important to maintain *in situ* image and text correspondence—that is, to preserve the pixel locations on the image for every word recognized with OCR. Search systems then indexed this text and location data. Users looking for a particular query term could be taken directly to the pages of books where that query term appeared, with their result highlighted on the image of the page. The use of the original page image is critical to providing users with access to the original document, because even the most sophisticated OCR software will make mistakes in the transcription process. Providing the recognized text as an

1. In this paper we discuss a number of software packages. To assist the reader in distinguishing between literature citations and software, we will reference and cite them separately. Literature references are cited numerically, while software references are cited alphabetically.

WWW 2012 Companion, April 16–20, 2012, Lyon, France.
ACM 978-1-4503-1230-1/12/04.

"invisible layer" on top of the original page image allows the user to navigate the book using query terms, but then to read the book from the image in its original form. This small shift in the use of OCR from transcription to navigation has opened up the unprecedented ability to navigate millions of books in an instant; a task that would have taken a lifetime of manual labour.

This shift has not been reflected in OMR systems, however. Most OMR software available today is used primarily as transcription tools for circumventing manual entry into a notation editor. They are largely based around a graphical user interface (GUI), designed for a single user on a single workstation. The export formats they use are suitable for import into a music notation editor but discard any information about the location of the notation on the original page image. To bring document image navigation using automatic transcription into a musical context, OMR software must be re-designed to focus on methods of high-volume image throughput, and on document output formats designed to maintain fine-grained correspondence between images and their transcriptions.

Our initial investigations into scaling OMR have involved deconstructing the traditional OMR process into separable tasks that together form a workflow, with the hopes of identifying places where the entire process can be made more efficient. Dissecting the OMR process allows each task to be identified and analyzed for any potential gains in workflow throughput, either by completely automating the task or by identifying places where automation could be used in conjunction with human supervision.

The human supervision component of any digitization and recognition workflow is always the most expensive step. Humans require salaries and workspace, and do not perform as well as a computer when performing tasks that are easily automated. They typically get tired, bored, and sloppy if presented with a task that requires too much repetition and concentration, and can only work for a certain number of hours per day. However, humans are critical components for performing quality control in the process, correcting the inevitable errors that automated systems make and ensuring these errors do not compound themselves in subsequent workflow steps. By focusing human intervention on only the tasks that a computer cannot do, or cannot do well, we reduce the amount of error in the final product while maintaining a level of quality and efficiency beyond simple automatic recognition. We have identified three broad strategies for optimizing human intervention:

- Use adaptive systems that learn
- Distribute the tasks across many humans
- Optimize the task for efficient performance

Adaptive Optical Music Recognition (AOMR)[6] has been proposed as a means of dealing with the large number and varying styles of music notation symbols. However, it has recently been recognized as a means of reducing costs in a digitization workflow [7]. By employing constantly learning software, the computer system requires less human intervention as it is given more exemplar pages.

As mentioned earlier, current OMR software is designed for a single user on a single workstation. This severely limits the number of people that can work on a given document at any time. Decoupling the software from the workstation and placing it on a remote server opens the door for many optimizations. Users can log in and perform their OMR tasks from any computer equipped with a modern web browser. Distributed proofreading and correction techniques, such as those employed by the IMPACT project [4], can be potentially performed by anyone, anywhere in the world. While libraries and archives may choose to still employ people to perform this work, the tasks may be distributed among specialists, and any member of a given pool of workers may perform a given task.

Finally, web-based software and distributed proofreading opens up the possibility of optimizing the actual tasks. While many users may be hesitant at the prospect of re-editing an entire score, it may be possible to "chunk" up correction or verification tasks into much smaller units of work which can then be distributed over a large number of users. The ReCAPTCHA project [8] has used this to great effect, asking a user to transcribe two OCR words to prove they are human as a mechanism for preventing automated spam bots. Other initiatives, like MajorMinor [9], have turned large-scale data collection and verification into a game where participants are rewarded points for their work. By examining the task and creating a highly optimized method of performing it, we may achieve acceptable overall throughput while still maintaining a level of quality control better than that of purely automated recognition.

There have been several previous attempts at building large-scale search and retrieval systems from OMR sources. We will examine these projects in the following section.

3. PREVIOUS WORK

The PROBADO Music Project [10,11] is perhaps the largest and longest-running project incorporating large-scale OMR for use in search systems. This project seeks to provide a unified interface for retrieving symbolic and audio representations of music pieces. As of October 2010 their dataset consisted of 50,000 pages from 292 books [12]. The content of their dataset is music printed in common Western notation in a variety of genres and instrumentations, including opera, symphonic works, and Classical and Romantic piano music.

The primary goal of the PROBADO project is to allow symbolic and audio synchronization, providing users with the ability to navigate a score and hear the audio, or navigate the audio and jump to its corresponding position in the score. For this use case, they have demonstrated that their OMR results do not need to be highly accurate to produce acceptable results. Their technique generates MIDI files from OMR, which are then rendered to an audio representation. This audio representation is then aligned with different instrumental recordings of the work. Despite the noisy nature of the underlying recognition, there is still enough information in the rendered MIDI audio to accurately align them. In [13] the authors states that they have systems in place to correct errors that most negatively affect synchronization, but make no mention of the extent to which all recognition errors are corrected.

Viro [14] describes a system used to perform OMR on the digitized scores held by the International Music Score Library Project (IMSLP), also known as the Petrucci Music Library. His project, Peachnote, uses off-the-shelf commercial and open-source OMR software to produce searchable representations of more than one million page images from 65,000 scores contained in IMSLP. These were then indexed as *n*-grams and made available for analysis via the Google *n*-grams viewer.

Finally, Choudhury et al. [15,16] report on developing a system at Johns Hopkins University for performing OMR on the entire Lester S. Levy Sheet Music Collection. The papers describe in how a large-scale OMR system might be implemented, but no workable prototype of their workflow system was produced.

4. THE *LIBER USUALIS* PROTOTYPE

To begin our investigations, we chose to build a complete prototype system; developing tools and processes necessary to bring a single large document from the point of image capture through to a basic search web application. We used the *Liber Usualis* [17], a service book produced by the Roman Catholic church and an important source of chant notation, as the basis for our first large-scale production. The *Liber Usualis* fits a number of criteria. It is a large book (2,340 page images), which makes it suitable for observing inefficiencies that might not be noticeable with fewer pages. It has been produced mechanically (i.e., printed), and so the layout and symbols are uniform across the whole book (unlike hand-produced sources). It contains a mixture of text (lyrics, readings, instructions, etc.) and music notation, which provides a real-world example of a mixed-content source in which the software must identify and separate the musical content from non-musical content. Finally, the music is monophonic (single-voiced) and uses only a very small set of articulation marks, which reduces the complexity of the musical notation and resulting digital representation.

Perhaps the most notable feature of the *Liber Usualis* is the system of music notation it uses. The music is expressed using square-note neume notation, an ancient system dating back to around the 12th Century. There are many similarities to modern notation. It uses a staff with lines (although it uses only four staff lines, and not the modern five), and has clefs that specify which pitch a line represents ("C" and "F" clefs are used). However, it differs from modern notation in that it combines notes into groups, called neumes. Since this type of music is used for chanting liturgical texts, these groupings are used as guides for syllabification, where a single syllable may get two, three, or more pitches. There are a number of standard neumes, each with their own name, but custom groupings of neumes may also be formed and fall under the general "compound" neume type. Some examples of these are given in Figure 1.

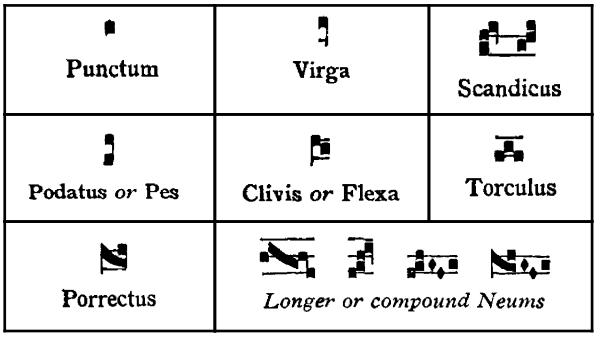

Figure 1: Selected neume shapes and names

4.1 Image Processing

The original source for the *Liber Usualis* was a PDF file [18] made available by the Canons Regular of St. John Cantius. The page images were exported from PDF using Adobe Acrobat Professional as 500dpi TIFFs, resulting in 2,340 image files. The source file had been previously processed for OCR using ABBYY FineReader [A], and so the resulting image files were pre-binarized using the built-in ABBYY binarization tools.

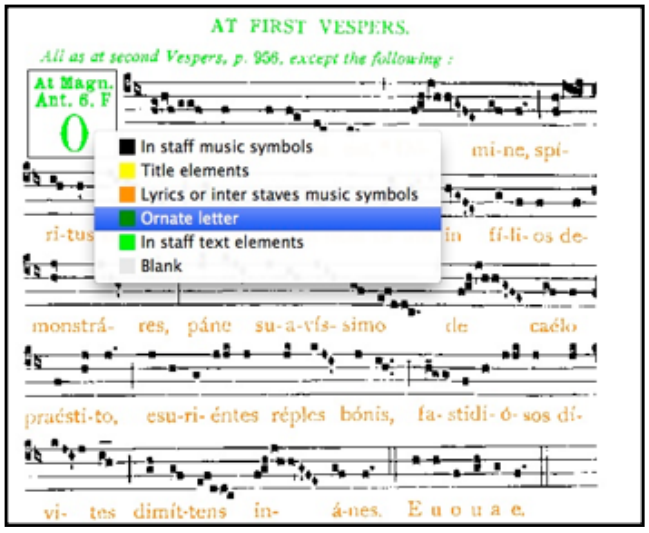

Figure 2: Layout analysis in Aruspix

4.2 Layout Analysis

Once the images were exported, we performed automated page layout analysis on them using a modified version of Aruspix [B], an OMR system designed for early music. This layout analysis detects five different types of page elements: music, titles, lyrics, ornate letters, and other text elements. A sixth option, "blank," is generally used for image artifacts like borders and creases that may appear as black pixels on the image but are not part of the content of the page. We wrote a Python script to run Aruspix on every image without human intervention. After all the pages were automatically segmented we had a human confirm and correct any errors in the layout analysis. The median correction time per page was 77 seconds, with the majority of pages taking between 30 and 130 seconds. The result was an image that could be segmented into separate layers containing textual or musical content exclusively (Figure 2). The layers containing musical notation were sent to OMR software, while the text layers were sent to OCR software.

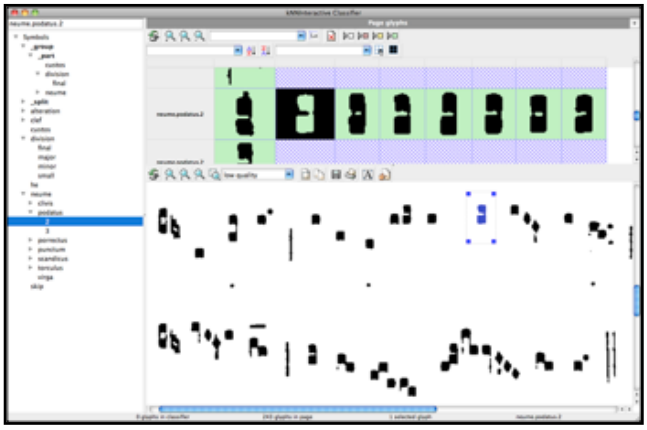

Figure 3: The Gamera classifier interface

4.3 OMR

We used the Gamera [19, C] software framework for performing OMR in this stage of the workflow. Images containing only music notation were processed to remove the staff lines using the Gamera Musicstaves toolkit [D]. The result was an image that had only neume shapes on it (Figure 3). These shapes were then

automatically classified using a pre-trained Gamera classifier. The automated classification for each page was then verified and corrected by musicians familiar with square-note notation. The median correction time per page was 11 minutes, with the majority of pages taking between 7 and 16 minutes.

4.4 OCR

Page images with only textual content were sent to a separate recognition process using the OCRopus OCR software [E]. As a proof-of-concept we applied only minimal effort to create an accurate transcription of the text. The raw recognized text was processed through an automatic spelling corrector trained on a dictionary of liturgical Latin words. For lyrics, syllables with dashes between them were automatically joined to form a single word. Much like the OMR stage, the OCR output for each line of text was correlated with a region on the image in order to allow us to highlight search results *in situ.*

4.5 Encoding and Correction

After the music recognition was complete we re-introduced the staff lines onto the image. We identified the clef shape and position for each staff, and correlated it with a staff line. The initial pitch for each neume was identified based on its position relative to the staff line and the clef, and the neume class name was used to identify each subsequent pitch in that neume.

One interesting and novel feature of this project was the use of class names to assist in recognizing the pitch content of a neume group (Figure 4). Class names for the shapes were constructed by using the pitch contour of the neume. For complex compound neumes, direction information was encoded in the class name. The result was a set of class names where the specific pitch contour of a neume could be reconstructed from knowing only its starting pitch. For example, a three-note torculus belonging to the torculus.3.2 class and starting on a *g* would outline the notes *g, b, a.* Full details of this work, including performance evaluations, can be found in [20].

Neume Class Name	Shape
`neume.torculus.3.2`	
`neume.scandicus.2.2.2.he`	
`neume.compound.u2.u2.d2.u2`	

Figure 4: Some neume shapes and class names

To store the recognized music notation we used the Music Encoding Initiative (MEI) format [21]. This format is designed to provide a common platform for encoding many different types of music notation, including neume notation. Inspired by the Text Encoding Initiative (TEI), MEI includes extensive methods for doing in-depth document description and encoding. Typically, symbolic music formats focus specifically on encoding one type of music notation (e.g., common Western notation, mensural, or neume notation), leading to a fragmented field where some systems support only certain types of notation. This is true of the more popular notation encoding types, specifically MusicXML which only supports common Western notation. Since we are designing this system with the goal of supporting different types of notation, we considered MEI to be a more comprehensive solution, allowing us to build a common set of tools for working with this format while reducing the number of formats that we need to support in the future.

For OMR applications, MEI has the ability to store both neume notation and the pixel-based image location references for all musical symbols [22]. This allows us to store the output of the OMR software in a symbolic notation format, while still maintaining image-to-symbol correspondence. The output of the Gamera OMR process was encoded using the PyMEI library [F].

For a final pitch verification and correction step the MEI and image files were then re-opened in Aruspix using a custom-designed graphical neume editor. Upon opening the page, the human corrector could see the original page image and the automatically recognized musical output rendered as editable notation. The human then corrected any mis-recognized pitches and saved the output again in MEI.

4.6 Search System

To provide a very basic pitch search, the neume pitches in each MEI document were indexed using simple *n*-grams, from 2 to 10 grams. We used a variation of the technique presented by Downie [23], storing absolute value of each *n*–gram, as well as *n*–gram values for contour, interval, and component neume names (Figure 5). By pre-computing these indexes we converted a musical query to a simple string-matching task—something that most modern search engines are highly optimized to perform.

```
contour: “dduurr”
intervals: “d2_d2_u2_u2_r_r”
location: [{'width': 407, 'ulx': 257, 'uly':
1459, 'height': 65}]
neumes: “punctum_clivis_podatus_punctum_punctum”
pagen: 157
pnames: edcdeee
semitones: -2_-2_2_2_0_0
```

Figure 5: Sample index entry

We initially used ElasticSearch [G] as our query management software. This software responded to updates to our CouchDB [H] databases and automatically updated its index as new content was added, facilitating very fast lookups over large indexes. While this setup provided the desired functionality, ElasticSearch crashed regularly and needed constant monitoring. To remedy these problems we replaced both the CouchDB and ElasticSearch components with a Solr [I] instance. Our current Solr instance indexes 3,006,964 unique *n*-gram documents in a structure identical to that originally stored in CouchDB. Solr has provided us with a simple, stable, and reliable architecture for our search system. Indexing the *Liber* corpus was performed with a custom Python script, and took one hour to complete. Uncached query times (i.e., queries that are performed after Solr’s query cache has been flushed) are typically under 10ms.

A significant disadvantage to this method of pitch search is that users cannot search on features of the notation that are not indexed, since they are not querying the MEI files directly. This is an area we feel is lacking in the current MIR research: How to perform un-indexed queries on a large corpus of symbolic notation and have it return a result in a timely fashion. While some authors have proposed optimizations to the indexing process [24,25,26], indexing purposefully yet significantly reduces the number of dimensions available for querying. We hope to address this problem in future work.

4.7 Web Application

In order to allow users to interact with the recognized and encoded *Liber Usualis* we built a prototype web application [27] that provides an interface to view the original document images, as well as to search and browse the recognized content (Figure 6). The web application serves the HTML and JavaScript front-end to the users, manages the server-side components for the image viewer, and functions as a proxy for search requests to our Solr index. This component was initially implemented using the Django framework [J], but because Django's database interface functionality was not needed it was subsequently ported to the much simpler Tornado framework [K].

We used a modified version of the Diva.js large document image viewer software [28,29,L] for providing the web application front-end to the search system. Using this software, we display all page images as a single scrollable entity on a page. The Diva.js viewer optimizes the viewing so that only the portion of the document that is being displayed in the browser is downloaded at any given time. This results in a very fast interface for navigating the document, even over wireless connections.

To provide *in situ* search results, we used the co-ordinates stored during the *n*-gram indexing to highlight the results of a pitch sequence search on the pages. Highlighting a region on an image uses HTML block elements for defining the location and size of the search result, with CSS used to provide colour and transparency.

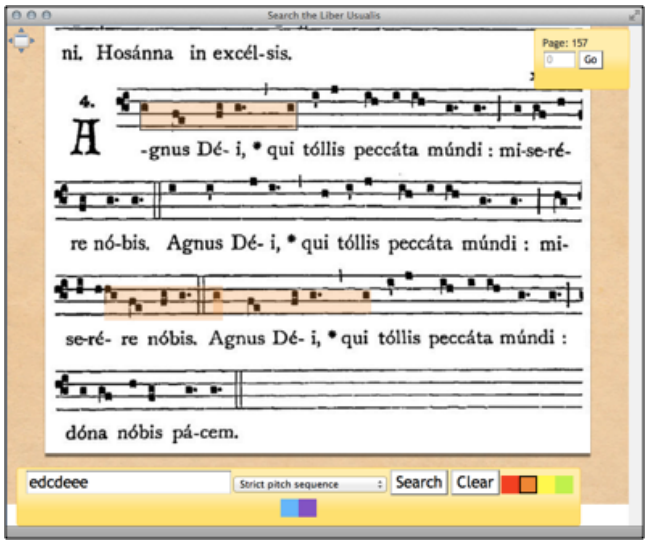

Figure 6: *Liber Usualis* web application interface with a highlighted search query ("edcdeee")

5. DISCUSSION AND FUTURE WORK

Printed music documents contain a wealth of musical information; indeed, much of the world's musical output over the last few centuries can only be found in physical form. While libraries and archives are diligently digitizing their documents and placing them online, users must still rely on traditional cataloguing data to navigate these document collections. To create the next generation of digital music document libraries, OMR software must be developed to manage high volumes of music document pages in an efficient manner though full text and music search.

As part of the SIMSSA project we are developing new enhancements to the traditional OMR process. We are employing Web 2.0 technologies to facilitate the creation of an online OMR web application to further allow the distribution of recognition and correction tasks to anyone with a web browser and an Internet connection. By decoupling the OMR process from a physical location, we hope to leverage the expertise of users around the world to assist libraries and archives in processing their collections.

Another benefit of placing the recognition process "in the cloud" is that we may then use the transcriptions provided by humans as collections of ground truth data for further improving adaptive OMR software. Maintaining image and transcription correspondence for a large number of musical documents will allow new types of symbol recognition techniques to be tested for performance and accuracy, and any improvements to the OMR process may be integrated into the online web application immediately, without having to distribute updates to all users.

Finally, we are investigating how users may navigate these document collections. One of the restrictions mentioned earlier is that the practice of indexing symbolic notation significantly reduces the types of queries a user can ask a retrieval system. In most current systems, both the query and the underlying musical documents need to be reduced to some form of text representation. This restricts query interfaces to either providing only very basic string-based matches or requiring a complex query language to notate musical commonplaces like chords, dynamics or multiple voices. By integrating sophisticated music analytical software like music21 [M] and Humdrum [N] with large search systems, such as Solr, or data processing systems, such as Hadoop [O], we hope to provide users with more sophisticated query systems while simultaneously enabling large-scale corpus analysis.

6. CONCLUSION

The technology for providing page-level access to musical materials is still in its infancy. In this paper we have presented our work on a prototype system for efficient access. Through this prototype we are exploring new methods of scaling OMR to meet the needs of high-throughput applications, new tools, and interfaces for exploring digital musical documents, and best practices for libraries and archives interested in making their collections available online. Our hope is that these experiences and tools will help in creating next-generation globally accessible digital music libraries.

7. ACKNOWLEDGEMENTS

The authors would like to thank our outstanding research and development team: Greg Burlet, Remi Chiu, Mahtab Ghamsari, Jamie Klassen, Saining Li, Wendy Liu, Mikaela Miller, Catherine Motuz, Laura Osterlund, Alastair Porter, Caylin Smith, and Jessica Thompson. This work was sponsored by the Social Sciences and Humanities Research Council (SSHRC) of Canada.

8. REFERENCES

[1] Vincent, L. 2007. Google Book Search: Document understanding on a massive scale. In *Proc. Intl. Conf. Document Analysis and Recognition.* 819–23.

[2] Christensen, H. 2011. HathiTrust: A research library at web scale. *Library Resources & Technical Services* 55 (2). 93–102.

[3] OpenLibrary Project. http://openlibrary.org.

[4] Balk, H., L. Ploeger. 2009. IMPACT: working together to address the challenges involving mass digitization of

historical printed text. *OCLC Systems & Services* 25 (4). 233–48.

[5] SIMSSA. http://simssa.ca.

[6] Fujinaga, I. 1996. Adaptive optical music recognition. Ph.D. diss. McGill University.

[7] Pugin, L., J. A. Burgoyne, and I. Fujinaga. 2007. MAP adaptation to improve optical music recognition of early music documents using hidden Markov models. In *Proc. Intl. Conf. Music Information Retrieval*. Vienna, AT. 513–6.

[8] Ahn, L., B. Maurer, C. McMillen, D. Abraham, M. Blum. 2008. reCAPTCHA: Human-based character recognition via web security measures. *Science* 321 (5895). 1465–8.

[9] Mandel, M., D. Ellis. 2008. A web-based game for collecting music metadata. *J. New Music Research*. 37 (2). 151–65.

[10] Diet, J., F. Kurth. 2007. The PROBADO music repository at the Bavarian State library. In *Proc. Intl. Conf. Music Information Retrieval.* Vienna, AT.

[11] Kurth, F., M. Müller, C. Fremerey, Y. Chang, M. Clausen. 2007. Automated synchronization of scanned sheet music with audio recordings. In *Proc. Intl. Conf. Music Information Retrieval.* Vienna, AT.

[12] Fremerey, C. 2010. Automatic organization of digital music documents: Sheet music and audio. PhD diss., U. Bonn.

[13] Damm, D., C. Fremerey, V. Thomas, M. Clausen. 2011. A demonstration of the PROBADO music system. Presented at *Int. Conf. Music Information Retrieval*. Miami, FL.

[14] Viro, V. 2011. Peachnote: Music score search and analysis platform. In *Proc. Intl. Conf. Music Information Retrieval.* Miami, FL.

[15] Choudhury, G., M. Droetboom, T. DiLauro, I. Fujinaga, B. Harrington. 2000. Optical music recognition within a large-scale digitization project. In *Proc. Intl. Conf. Music Information Retrieval*. Plymouth, MA.

[16] Choudhury, G. S., C. Requardt, I. Fujinaga, T. DiLauro, E. W. Brown, J. W. Warner, and B. Harrington. 2000. Digital workflow management: The Lester S. Levy digitized collection of sheet music. *First Monday* 5 (6).

[17] Catholic Church. 1963. *The Liber Usualis, with introduction and rubrics in English.* Tournai, Belgium: Desclée.

[18] Sancta Missa. http://www.sanctamissa.org /en/music/ gregorian-chant/ *choir/liber-usualis-1961.pdf.*

[19] Droettboom, M., K. MacMillan, and I. Fujinaga. 2003. The Gamera framework for building custom recognition systems.
Proceedings of the Symposium on Document Image Understanding Technologies. 275–86.

[20] Vigliensoni, G., J. A. Burgoyne, A. Hankinson, and I. Fujinaga. 2011. Automatic pitch detection in printed square-note notation. *Proc. Intl. Conf. Music Information Retrieval Conference*. Miami, FL. 423–8.

[21] Roland, P. 2009. MEI as an editorial music data format. In *Digitale Edition zwischen Experiment und Standardisierung,* eds. P. Stadler, and J.Veit, 175–94.Tübingen: Max Niemeyer.

[22] Hankinson, A., L. Pugin, and I. Fujinaga. 2010. An interchange format for optical music recognition applications. In *Proc. Intl. Conf. Music Information Retrieval*, Utrecht, NL.

[23] Downie, J. S. 1999. Evaluating a simple approach to music information retrieval: Conceiving melodic *n*-grams as text. PhD diss.,U. Western Ontario.

[24] Typke, R., A. Walczak-Typke. 2009. Some vantage indexing approaches for rhythmic and melodic search. *Musicae Scientiae Discussion Forum* 4B. 201–34.

[25] Typke, R., F. Wiering, R. Veltkamp. 2007. Transportation distances and human perception of melodic similarity. *Musicae Scientiae Discussion Forum* 4A. 153–81.

[26] Lemström, K., N. Mikkilä, V. Mäkinen. 2010. Filtering methods for content-based retrieval on indexed symbolic music databases. *Information Retrieval* 13. 1–21.

[27] Online *Liber Usualis*. http://ddmal.music.mcgill.ca/liber

[28] Hankinson, A., L. Pugin, and I. Fujinaga. 2009. Interfaces for document representation in digital music libraries.In *Proc. Intl. Conf. Music Information Retrieval*, Kobe, JP.

[29] Hankinson, A., W. Liu, L. Pugin, I. Fujinaga. 2011. Diva.js: A continuous document viewing interface. *Code4Lib Journal* 14.

9. SOFTWARE

[A] ABBYY FineReader. http://www.abbyy.com/

[B] Aruspix. http://aruspix.net/

[C] Gamera. http://gamera.informatik.hsnr.de/

[D] Musicstaves Toolkit. http://gamera.informatik.hsnr.de/ addons/musicstaves/

[E] OCRopus. http://code.google.com/p/ocropus/

[F] PyMEI. https://github.com/ahankinson/pymei

[G] ElasticSearch. http://www.elasticsearch.org/

[H] CouchDB. http://couchdb.apache.org/

[I] Solr. http://lucene.apache.org/solr/

[J] Django. https://www.djangoproject.com/

[K] Tornado. http://www.tornadoweb.org/

[L] Diva.js. http://ddmal.music.mcgill.ca/diva/

[M] Music21. http://mit.edu/music21/

[N] Humdrum. http://www.music-cog.ohio-state.edu/Humdrum/

[O] Hadoop. http://hadoop.apache.org/