AN INTERCHANGE FORMAT FOR OPTICAL MUSIC RECOGNITION APPLICATIONS

Andrew Hankinson[1] **Laurent Pugin**[2] **Ichiro Fujinaga**[1]
[1]Centre for Interdisciplinary Research in Music Media and Technology (CIRMMT)
Schulich School of Music, McGill University
[2]RISM Switzerland & Geneva University
andrew.hankinson@mail.mcgill.ca, lxpugin@gmail.com, ich@music.mcgill.ca

ABSTRACT

Page appearance and layout for music notation is a critical component of the overall musical information contained in a document. To capture and transfer this information, we outline an interchange format for OMR applications, the OMR Interchange Package (OIP) format, which is designed to allow layout information and page images to be preserved and transferred along with semantic musical content. We identify a number of uses for this format that can enhance digital representations of music, and introduce a novel idea for distributed optical music recognition system based on this format.

1. INTRODUCTION

Page appearance and layout for music notation is a critical component of the overall musical information contained in a document. For example, musically semantic information, such as note duration, is often visually augmented by adjusting horizontal spacing to reflect a spatial representation of note duration [1]. Some scholars infer geographical origin or time period based on note shapes, or even, in the case of handwritten manuscripts, by the particular "hand" of a scribe [2, 3]. The layout may also reveal some subtle intent of the composer, especially in sketches and autograph manuscripts [4].

To date, however, there has been little effort to attempt to preserve this information when a page is scanned and processed by optical music recognition (OMR) software. This presents several opportunities for improvement. By maintaining a direct relationship between recognized musical symbols and the original image it was extracted from, we contend that musicians and music scholars will be better able to understand and interpret digital facsimiles of musical documents while simultaneously providing the ability to index, search, and retrieve these documents.

For OMR researchers, this also presents an opportunity to build large global ground-truth datasets. By maintaining the relationship between the graphical representation and the semantic interpretation of a musical symbol, we can build sets of training data which exemplar-based adaptive supervised-learning software can use to train and test its recognition models. Furthermore, by allowing for these datasets to be shared between different adaptive OMR platforms, we can take advantage of work done by others who have created different datasets to further improve recognition software. This is discussed further in Section 4.

In this paper, we present the OMR Interchange Package (OIP) format, a common interchange format for OMR applications that bundles notation, images, and metadata together in a single file. Work on this format was inspired by functionality present in large, established digitization projects, most notably Google Books and the Internet Archive. These projects use file formats designed to preserve layout information in textual materials. We discuss two such formats, hOCR and DjVu, and examine them for ideas of how we might construct a similar music notation-specific format.

Rather than build a completely separate set of specifications, the OIP format combines established standards into an application profile—that is, we provide specifications on how these standards should be combined. These standards concern music, image, and metadata encoding formats, contained within an established standard for packaging and serializing these files into a single file, for easy transport across multiple systems. By taking an application profile approach, instead of establishing a new, monolithic standard, we hope to take advantage of existing software to manipulate component files, e.g., reading and writing images, and delegate the maintenance and improvement of the component standards to their respective communities.

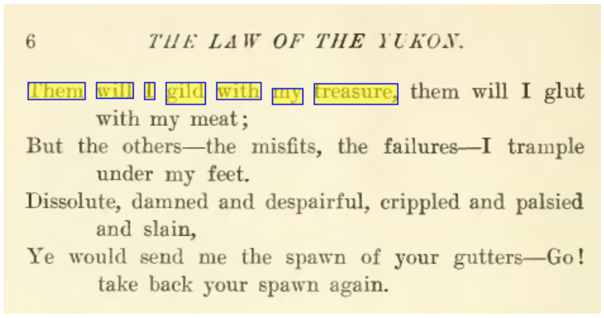
One of the goals for developing the OIP format is to provide a mechanism for interchange between different elements in an OMR digitization workflow, from capture through recognition and into any number of potential uses. Specific design considerations were made to ensure that non-common practice notation systems are accommodated, to allow for encoding earlier musical print and manuscript sources.

2. BACKGROUND

2.1 Optical Character Recognition

The Google Books project [5] and the Internet Archive [6] are industrial-scale initiatives to convert physical textual items, e.g., books, magazines, and newspapers, to searchable digital representations. As items in these collections are digitized, their page images are processed by OCR software, extracting the textual content, and facilitating full-text searching of their collections.

Within the OCR workflow, the precise location on the page where a word occurs is saved through the use of a bounding box that defines a region around the word. When the words on the page are converted to searchable text, the bounding box coordinates are stored, along with the word itself. In some cases, similar coordinates can be stored to outline higher-level page elements such as lines, columns, or paragraphs. Figure 1 shows an example from the Internet Archive of a page image returned from a full-text search with the phrase “Them will I gild with my treasure” highlighted in reference to its position on the original page scan.



6 THE LAW OF THE YUKON.

Them will I gild with my treasure, them will I glut
with my meat;
But the others—the misfits, the failures—I trample
under my feet.
Dissolute, damned and despairful, crippled and palsied
and slain,
Ye would send me the spawn of your gutters—Go!
take back your spawn again.

Figure 1: Document with search terms highlighted *in situ*. (Source: Internet Archive)

In contrast, we can find little evidence to suggest similar techniques are in widespread use for databases of music documents. Instead, collections either choose to simply display the page image with no transcription of the source available (e.g., [7, 8]), or transcribe the content into a searchable and manipulable digital format without reference to the original page layout (e.g., [9]). For music documents, where the layout of the symbols can play a critical role in determining the intended interpretation of the music, we posit that a hybrid approach is needed, similar to that demonstrated by Google Books or the Internet Archive.

Critical to the development of these systems is a common standard that allows various systems in an OMR workflow to capture and preserve images, layout, and music semantics. To help inform our development of such a standard, we identified formats used in the textual domain for encoding layout information: The hOCR format, developed as an output format for the Google-sponsored OCRopus document analysis software, and DjVu, a third-party document imaging solution adopted by the Internet Archive for displaying its digitized texts.

2.1.1 hOCR

hOCR [10] is a format that uses standard HTML tags, but embeds OCR-specific information that can be read and manipulated by other OCR software. According to the authors of the hOCR specification, it can be used to encode “layout information, character confidences, bounding boxes, and style information” [11]. For generic HTML rendering software, like a web browser, the OCR-specific information is ignored and the page is rendered without interference.

For the developers of hOCR, HTML was preferred over the definition of a new XML format since the HTML specification already contains many tags for defining document elements, such as headings, tables, paragraphs, and page divisions. Furthermore, the files can be viewed, manipulated, and processed with a wide range of existing tools, such as browsers, editors, converters, and indexers.

To encode information about a page layout, hOCR uses the “class” and “title” attributes of HTML tags. For example, a bounding box outlining a paragraph may be defined as:

```
   <div class=”ocr_par” id=”par_7”
title=”bbox 313 324 733 652”>
    ...paragraph text...
   </div>
```

Figure 2: hOCR format defining a paragraph bounding box

The bounding box is given as two sets of pixel co-ordinates corresponding to the upper-left and lower-right corners of the box, relative to the upper-left corner of the image.

Page images corresponding to the text output are linked from the hOCR document with either a local path name or an HTTP URL. The identity and integrity of the image file can be verified by embedding the MD5 checksum of the image file in the hOCR file.

2.1.2 DjVu

DjVu is primarily designed as a highly efficient method of compressing and transferring images and documents. Included in its specification, however, is the ability to include a “hidden” text layer within a binary DjVu file.

The DjVu format specification [12] defines seven different types of document “zones,” each featuring a bounding box defined by an offset co-ordinate from a previously defined zone and a given width and height. These zones can define boundaries for pages, columns, regions, paragraphs, lines, words, or characters. Text is encoded as UTF-8.

2.2 Music Applications

hOCR and DjVu are not the only formats that can provide positional information about text. The popular PDF standard allows for this functionality as well. They serve, however, as examples of existing formats in the textual domain from which we can begin to discuss similar approaches in the musical domain.

To begin building our standard, we define five basic criteria that the OIP format should conform to:

2.2.1 Must be self-contained

Files conforming to the OIP format should be self-contained in a single file. The choices here are between defining a unique binary format, as the DjVu format does, or allowing multiple files to be packaged as a single file.

2.2.2 Must encapsulate multi-page documents

Both hOCR and DjVu encode multiple pages in a single file. hOCR provides only the textual content of those pages and links to externally stored images, while DjVu stores both image and content for multiple pages within a single file.

2.2.3 Must encapsulate notation, images, and metadata

For each page in the document our format must include a page image, the notation content, and, if available, any other metadata about that page. Here, the music domain requires a different approach than the text domain, owing largely to the complexity of encoding music notation over encoding text. In Section 3, we discuss the specific standards chosen for this criteria.

2.2.4 Must use existing standards

Drawing largely on the arguments made by the hOCR developers to justify their use of HTML over creating a new format [10], we specify that, wherever possible, existing standards must be used in preference to creating one. This is especially true for encoding notation, where new formats are introduced every few years, often designed to meet very specific needs, and fall out of use within a few years of being introduced. By using existing standards, we hope to ensure a broader support community beyond our specific application.

2.2.5 Must allow extended information

Beyond the required notation, images, and metadata storage, we see the OIP format as a general-purpose container for storing any extra information about the page content. However, this extra information should be opaque to clients that do not support it, and should not interfere with their ability to read and write OIP files. For example, a specific application could save extended colour-space information about an image in the OIP, available to applications that can use it, but ignored by clients that cannot use it.

3. FORMAT SPECIFICATION

As discussed in the previous sections, we have chosen to combine existing standards into an application profile. In this section we will discuss our specific choices of standards and how they should be combined to create an OIP-compliant file. In the interests of space we will specifically avoid any in-depth explanation about the component standards themselves, since they are freely available for consultation.

3.1 Packaging

An OIP file is, at its most basic representation, a collection of files and folders serialized as a single file. Rather than simply allowing an *ad hoc* method of bundling these files and folders together, we chose to use a very minimal standard for organizing the content of these files.

There are several ways to approach this problem. One type of solution is exemplified by formats such as the Metadata Encoding and Transmission Standard (METS). Data typically represented in binary formats (e.g., images) can be stored, for example, within an XML file by Base64 encoding. A single METS file containing many high-quality images could potentially be many gigabytes in size, however.

A second approach is the file bundle approach. This is used by many formats, including Microsoft's XML-based Office formats (e.g., DOCX) and the Java JAR format. These files are simple file and folder hierarchies containing component files, such as images or text files. They appear as a single file archive by using a well-known file archiving system (e.g., ZIP or TAR). Once these bundles have been uncompressed, read and write operations on the smaller component files can be done directly via the native file system and not on the single monolithic XML file.

The BagIt format is a lightweight file bundling specification. It was created and is maintained by the Library of Congress and the California Digital Library. It is currently in the process of becoming an IETF standard [13]. The name refers to a colloquial rendering of the Enclose and Deposit method [14], also known as the "bag it and tag it" method.

This format defines a simple hierarchy of files and folders, known as a "bag." These can be represented plainly on any computer system as standard files and folders, or they can be converted into a single file using ZIP or TAR packaging.

Minimally, one directory and two files must be present in every bag in order to be considered compliant to the standard. A data directory contains any arrangement of files or folders are stored. This is the bag's "payload." One of the required files is a bagit.txt file that simply stores the version of the BagIt specification to which that bag conforms and the character encoding used for the metadata files. The second required file is a manifest file listing checksums for each file within the data directory, helping to ensure the integrity and identity of each of the files in the bag. Other optional files are outlined in the BagIt specification [13].

```
<bag-directory>
      |- bagit.txt
      |- manifest-md5.txt
      |- [other optional bagit files]
      |- data
           |- [page 1]
           |      |- [image files]
           |      |- [notation files]
           |      |- [metadata files]
           |- [page 2]
           |      |- [image files]
           |      |- [notation files]
           |      |- [metadata files]
           |- [etc.]
```

Figure 3: A generalized OIP structure.

For the OIP format, we further specify a file hierarchy within the data directory of a bag. A folder is created in the data directory for each page in a multi-page document, allowing the format to accommodate documents of any size. In each page folder, we store files relating to this page. A generalized example of the OIP structure is given in Figure 3.

This does not create incompatibilities with the original BagIt specification, as there is no structure to which the data directory must conform. Software for processing BagIt files will guarantee the integrity and identity of each file in the bag without needing to understand the OIP format.

3.2 Notation

There are many file formats for encoding music notation, but for this specific application we require a format that can encode positional coordinates for every musical element on the page. This eliminates many traditional formats used for notation interchange, such as MIDI. The Notation Interchange File Format (NIFF) fits this requirement, but is no longer actively maintained and is considered an obsolete standard [15]. The SharpEye output format (MRO) [16] also encodes this information and is used by [17] to provide positioning information for musical elements. This format, however, is specifically designed for use with common Western notation (CWN), limiting its usefulness for older or alternative notation systems. MusicXML [18] and NeumesXML [19] focus respectively on CWN and neumed notation, limiting their applicability for a broad range of notation systems.

For OIPs, we recommend the use of the Music Encoding Initiative (MEI) format as a notation encoding scheme. MEI inherits many features of the Text Encoding Initiative (TEI), a format specifically designed for scholars representing original text sources in digital form. MEI can also adequately represent CWN as well as other notation systems [20].

MEI allows for bounding boxes, or “zones,” to be defined for a given image and identified with a unique ID. These *id*’s can then be attached to semantically defined musical elements in MEI. A brief example is shown in Figure 4.

```
<facsimile source="s2">
   <surface>
      <graphic xml:id="s2p1"
xlink:href="m000001719_0001.tif"/>
      <zone xml:id="s2p1z1" lrx="0"
lry="0" ulx="0" uly="0"/>
      <zone xml:id="s2p1z2" lrx="1"
lry="1" ulx="10" uly="10"/>
   </surface>
</facsimile>
<!-- ... -->
<measure n="1" xml:id="d1e656"
facs="s2p1z1"/>
```

Figure 4: A MEI-formatted example showing bounding box definitions.

In MEI, the `<graphic>` element defines a link to a page image, while subsequent `<zone>` elements outline regions of this image, identified with a unique `xml:id` attribute. These zones are then used later within the music notation markup, as illustrated in Figure 4 by the `<measure>` tag. It uses the `facs` attribute to link a defined bounding box to a measure definition. This attribute is available to all music notation elements.

3.3 Images

For image formats, we follow the guidelines given in [21] for musical master archival images. These guidelines recommend lossless file encoding formats such as TIFF or PNG for archival formats. While there is no technical reason for not using other formats such as lossy JPEG, we suggest lossless formats to maintain the highest possible image quality.

One issue we have not yet addressed is how to reconcile the differences between an original image and an image file that has been cropped, de-skewed, and prepared for processing by an OMR package. Since any geometric manipulation will affect the co-ordinates of the musical elements on the page, it would be difficult to automatically reconcile the position of musical elements in an original image, when the notation was extracted using a processed image. This becomes especially important when considering the OIP format as an interchange format between multiple OMR systems, each of which may use different image processing techniques, or even require that certain elements of an image be removed prior to recognition, such as staff lines.

To reconcile this, we specify that, at a minimum, an OIP should contain the original page image, and a page image that the OMR system used during the recognition stage prior to removing any musically relevant elements. The additional inclusion of any intermediary images used by OMR software is permitted, but not required. For an OIP that has been processed by multiple OMR packages, each package should save its source image and recognized notation in MEI.

3.4 Metadata

MEI has the facilities to capture bibliographic, analytic, and editorial metadata. There is also the possibility that other metadata can be captured and stored within the file hierarchy. While we do not require any further metadata beyond what can be captured in MEI, we do not prevent the inclusion of other files with metadata formats describing, for example, detailed image processing techniques, historical and archival information, or library-specific local information.

4. APPLICATIONS

We have formulated the OIP format as an interchange format between multiple elements of an OMR workflow, from digitization through recognition, and finally into a delivery format specifically designed to capture and transfer page layout along with the semantic music content. In this section, we identify three specific applications where OIP files can be implemented as a standardized format for constructing tools useful for music scholars and OMR research.

4.1 Diplomatic Facsimiles

While there is some disagreement on the actual definition of the term, we define diplomatic facsimile as "a visualization (on-screen or in print) from the digital transcription of a source artifact, such that it has the same semantic content as the source, and its glyphs and layout are similar to the original source" [22].

For notation styles outside of the CWN tradition, a diplomatic facsimile provides the ability to transcribe a musical source with its original layout and symbols, without interpreting it by using modern music notation symbols. Barton, Caldwell, and Jeavons provide an excellent overview of the importance of this distinction [23]. Diplomatic facsimiles also allow libraries and archives to withhold distribution of original images due to copyright restrictions, while simultaneously allowing scholars access to a faithful electronic reproduction of the original musical content, including precise positioning for each musical element in the document.

4.2 Online Music Document Databases

An online database of music documents, similar to Google Books or the Internet Archive's display of textual documents, could be constructed with OIP as a source format for these documents. In an OMR workflow, OIP files would serve as an interchange format between the OMR software and a database system designed to organize, index, and display these documents.

As mentioned in the introduction, music scholars often use visual cues in the layout of a page of music to determine how a piece of music might be performed, or where it came from. Viewing these documents in their original form, while still making them available for content-specific searching and indexing, would provide a valuable research tool for many music scholars.

Furthermore, an online music document database could highlight relevant musical phrases matching a user's query, displayed as an invisible layout on the original image. Advanced computer processing could potentially provide links between similar passages within, or across, musical pieces, allowing users to navigate a document by musical phrase.

4.3 Distributed Optical Music Recognition

The extent, variety, and variability of musical symbols pose a unique problem to optical music recognition software. These symbols encompass indications of pitch, duration, dynamics, tempo, or performer interpretation (e.g., turns and trills). Different printing practices or fonts also introduce variations in these shapes.

Adaptive OMR (AOMR) software attempts to account for this variability by using machine-learning methods for understanding and interpreting new shapes, or variations on known shapes. These systems are often trained using human annotators or correctors, who provide a system with the correct musical interpretation of a graphical shape [24].

This training process is often the most tedious and expensive part of the OMR process. Developing training sets of sufficient quantity and variety is an expensive and labour-intensive process. Similarly, a poorly trained recognition system will require more human intervention, leading to lower overall throughput for any digitization and recognition initiative. For large digitization projects, this can have a significant impact on the overall cost of digitizing these materials [25].

With a common interchange format, however, these data sets could be built cumulatively. As new pieces of music are recognized and corrected, this work can be saved and used to train other AOMR clients with no further intervention by a human annotator.

Perhaps more importantly, this concept can be used to build a distributed global network of AOMR clients. Sharing training data with other networked OMR clients would allow them to build their recognition models using data previously provided by other members of the network. For example, an archive that has provided a data set of examples from a 16th-Century Italian music printer can make this data set available immediately to other members of the network, allowing these clients to immediately re-train their recognition systems to take advantage of this new data and increase their accuracy on this particular repertoire.

5. CURRENT AND FUTURE WORK

To date, we have finished the initial release of an open source Python library for reading and writing BagIt files, available at [26]. This is part of a larger project to develop a distributed optical music recognition system, a networked collection of adaptive OMR clients.

6. REFERENCES

[1] Blostein, D., and L. Haken. 1991. Justification of printed music. *Communications of the ACM* 34 (3): 88–99.

[2] Warmington, F. 1999. A survey of scribal hands in the manuscripts. In *The treasury of Petrus Alamire: Music and art in Flemish court manuscripts 1500–1535*, ed. H. Kellman, 41–52. Amsterdam: Ludion Ghent.

[3] Luth, N. 2002. Automatic identification of music notations. In *Proceedings of the International Conference on Web Delivering of Music (WEDELMUSIC),* 203–10.

[4] Hall, P. 1997. *A view of Berg's Lulu through the autograph sources*. Berkeley: University of California Press.

[5] Google Books. http://books.google.com [Accessed 23 March 2010].

[6] Internet Archive. http://www.archive.org [Accessed 23 March 2010].

[7] Schubert Autographs. http://www.schubert-online.at/activpage/index_en.htm [Accessed 23 March 2010].

[8] Digital Image Archive of Medieval Music. http://www.diamm.ac.uk [Accessed 23 March 2010].

[9] The Computerized Mensural Music Editing Project. http://www.cmme.org [Accessed 23 March 2010].

[10] Breuel, T. M., and U. Kaiserslautern. 2007. The hOCR microformat for OCR workflow and results. In *Proceedings of the International Conference on Document Analysis and Recognition (ICDAR)*, 1063–7.

[11] hOCR-tools. http://code.google.com/p/hocr-tools [Accessed 23 March 2009].

[12] Celartem/Lizartech DjVu format reference. http://djvu.org/docs/DjVu3Spec.djvu [Accessed 23 March 2010].

[13] The BagIt File Packaging Format. https://svn.tools.ietf.org/html/draft-kunze-bagit-04 [Accessed 23 March 2010].

[14] Tabata, K., T. Okada, M. Nagamori, T. Sakaguchi, and S. Sugimoto. 2005. A collaboration model between archival systems to enhance the reliability of preservation by an enclose-and-deposit method. In *Proceedings of the International Web Archiving Workshop.*

[15] Castan, G. 2009. NIFF. http://www.music-notation.info/en/formats/NIFF.html. [Accessed 29 May 2010].

[16] Jones, G. OMR engine output file format. http://www.visiv.co.uk/tech-mro.htm [Accessed 29 May 2010].

[17] Kurth, F., D. Damm, C. Fremerey, M. Müller, and M. Clausen. 2008. A framework for managing multimodal digitized music collections. In *Research and advanced technology for digital libraries*, 334–45. Berlin: Springer.

[18] Good, M. 2009. Using MusicXML 2.0 for music editorial applications. In *Digitale Edition zwischen Experiment und Standardisierung*, ed. P. Stadler and J. Veit, 157–74. Tübingen: Max Niemeyer.

[19] Barton, L. 2002. The NEUMES project: Digital transcription of medieval chant manuscripts. In *Proceedings of the Web Delivering of Music (WEDELMUSIC)*, 211–8.

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

[21] Riley, J., and I. Fujinaga. 2003. Recommended best practices for digital image capture of musical scores. *OCLC Systems and Services* 19 (2): 62–9.

[22] Glossary of terms used by the NEUMES project. http://www.scribeserver.com/NEUMES/help/glossary.htm [Accessed 23 March 2010].

[23] Barton, L., J. Caldwell, and P. Jeavons. 2005. E-library of medieval chant manuscript transcriptions. In *Proceedings of the Annual Joint Conference on Digital Libraries (JCDL)*, 320–39.

[24] Fujinaga, I. 1997. *Adaptive optical music recognition*. PhD diss., McGill University.

[25] Pugin, L., J. Burgoyne, and I. Fujinaga. 2007. Reducing costs for digitising early music with dynamic adaptation. In *Proceedings of the European Conference on Digital Libraries (ECDL)*, 471–4.

[26] Hankinson, A. 2010. PyBagIt 1.0 documentation. http://www.musiclibs.net/pybagit [Accessed 29 May 2010].